ARTIFICIAL INTELLIGENCE IN TRANSPORTATION INDUSTRY

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

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING WITH SPECIALIZATION IARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Submitted by:

VUPPUTURI BHARATH - 20BCS6586

GOTTIPALLI SAI PRASANTH - 20BCS6832

KETHURI AJAY – 20BCS6585

KOYYADA AKSHAY KUMAR – 20BCS6369

Under the Supervision of:

SHUBHANGI MISHRA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

APEX INSTITUE OF TECHNOLOGY

CHANDIGARH UNIVERSITY, GHARUAN, MOHALI - 140413, PUNJAB

Aug - Nov. 2023

BONAFIDE CERTIFICATE

I Certified that this project report **“ARTIFICIAL INTELLIGENCE IN TRANSPORT INDUSTRY”** is the Bonafide work of **VUPPUTURI BHARATH, GOTIPALLI SAI PRASANTH, KETHURI AJAY, KOYYADA AKSHAY KUMAR** who carried out the project work under my supervision.

SIGNATURE OF THE HOD	**SIGNATURE OF THE SUPERVISOR**
MR. AMAN KOUSHIK	**SUBHANGI MISHRA**
(HEAD OF THE DEPARTMENT**)**	**(**SUPERVISOR)
CSE - AIML	**(**Assistant Professor)
	(AIT – CSE)

Submitted for the project viva voce examination held on

INTERNAL EXAMINER **EXTERNAL EXAMINER**

TABLE OF CONTENTS

ABSTRACT

The integration of Artificial Intelligence (AI) in the transportation industry has ushered in a new era of mobility characterized by efficiency, safety, and sustainability. This abstract explores the multifaceted role of AI in revolutionizing various domains within transportation, ranging from autonomous vehicles and traffic management to logistics and predictive maintenance.

Driver fatigue monitoring system is a real-time system that can detect driver fatigue and distraction using computer vision approaches. In this paper, a new approach is introduced for driver hypervigilance (fatigue and distraction) detection based on the symptoms related to face and eye regions. In this method, face template matching and horizontal projection of top-half segment of the face image are used to extract hypervigilance symptoms from the face and eye, respectively. Head rotation is a symptom to detect distraction that is extracted from face region.

The extracted symptoms from the eye region are the percentage of eye closure, eyelid distance changes with respect to the normal eyelid distance, and eye closure rate. The first and second symptoms related to eye region are used for fatigue detection; the last one is used for distraction detection. In the proposed system, a fuzzy expert system combines the symptoms to estimate level of driver hypo-vigilance. There are three main contributions in the introduced method simple and efficient head rotation detection based on face template matching, adaptive symptom extraction from eye region without explicit eye detection, and normalizing and personalizing the extracted symptoms using a short training phase. These three contributions lead to develop an adaptive driver eye/face monitoring. Experiments show that the proposed system is relatively efficient for estimating driver fatigue and distraction.

ACKNOWLEDGEMENT

It gives us immense pleasure to express our deepest sense of gratitude and sincere thanks to our respected guide Subhangi Mishra (Assistant Professor), CSE- Artificial Intelligence, Chandigarh University, Mohali for his valuable guidance, encouragement and help for completing this work. Her useful suggestions for this whole work and cooperative behavior are sincerely acknowledged. We are also grateful to Dr. Shikha Gupta (Program Leader, CSE-AIML) for her constant support and guidance.

We also wish to express our indebtedness to our family members whose blessings and support always helped us to face the challenges ahead. We also wish to express thanks to all people who helped us in completion of this project.

VUPPUTURI BHARATH - 20BCS6586

GOTIPALLI SAI PRASANTH - 20BCS6832

KETHURI AJAY - 20BCS6585

KOYYADA AKSHAY KUMAR - 20BCS6369

CHAPTER -1
INTRODUCTION

Artificial intelligence is revolutionizing the way we travel and ensuring our safety on the roads. As technology continues to advance at a rapid pace, it is becoming increasingly important for the transport industry to keep up with the latest innovations. One such innovation is the use of AI-powered systems, which have the potential to transform the way we move around. From autonomous vehicles to traffic management systems, AI has already made significant strides in improving the efficiency and safety of transportation. But perhaps one of its most important applications is in the area of driver fatigue monitoring. Driver fatigue is a serious issue that affects millions of people every year. It can lead to accidents, injuries, and even fatalities. By using AI to detect signs of fatigue in drivers, we can prevent these tragedies from occurring and make our roads safer for everyone.

Improvement of public safety and the reduction of accidents are of the important goals of the Intelligent Transportation Systems (ITS). One of the most important factors in accidents, especially on Truck Drivers, is the driver fatigue and monotony. Fatigue reduces driver perceptions and decision-making capability to control the vehicle. Researches show that usually the driver is fatigued after 1 hour of driving. In the afternoon early hours, after eating lunch and at midnight, driver fatigue and drowsiness is much more than other times. In addition, drinking alcohol, drug addiction, and using hypnotic medicines can lead to loss of consciousness.

In different countries, different statistics were reported about accidents that happened due to driver fatigue and distraction. Generally, the main reason of about 20% of the crashes and 30% of fatal crashes is the driver drowsiness and lack of concentration. In single-vehicle crashes (accidents in which only one vehicle is damaged) or crashes involving heavy vehicles, up to 50% of accidents are related to driver Hypervigilance. According to the current studies, it is expected that the number of crashes will be reduced by 10%–20% using driver fatigue monitoring systems. The driver fatigue monitoring system is a real-time system that investigates the driver physical and mental condition based on the processing of driver face images. The driver state can be estimated from the eye closure, eyelid distance, blinking, gaze direction, yawning, and head rotation. This system will alarm in the hypervigilance states including fatigue and distraction. The major parts of the driver fatigue monitoring system are imaging, hardware platform, and the intelligent software.

1.1 PROBLEM DEFINITION

In this project, we try to implement different types of classifiers that helps to overcome the challenges of driver fatigue. Design and implement a Driver Fatigue Monitoring System that utilizes technology to detect and alert drivers when they are showing signs of fatigue or drowsiness while operating a vehicle. The primary goal is to enhance road safety by preventing accidents caused by driver fatigue.

1.2 PROJECT OVERVIEW

A computer vision system that can automatically detect driver fatigue in a real time video stream and then play an alarm if the driver appears to be fatigue or drowsy. This revolutionizes the vast transport system, preventing massive human and financial losses. In this program we check how long a person's eyes have been closed for. If the eyes have been closed for a long period i.e., beyond a certain threshold value, the program will alert the user by playing an alarm sound.

1.3 HARDWARE AND SOFTWARE REQUIREMENTS

HARDWARE SPECIFICATIONS:

- Processor – 64-bit eight-core, 2.5GHz per core.
- RAM – Minimum 4GB required.
- Hard Disk – SSD or HDD minimum 40GB free space required

SOFTWARE SPECIFICATIONS:

- Edition - Windows 10/11
- OS build 19043.1526
- Python installed – version 3.7 to 3.10
- OpenCV library Installed
- Any python compiler or system terminal

Chapter - 2

LITERATURE SURVEY

The driver fatigue monitoring systems can be divided into two general categories. In one category, driver fatigue and distraction is detected only by processing of eye region. There are many researches based on this approach. The main reason of this large number of researches is that the main symptoms of fatigue and distraction appear in the driver eyes.

Moreover, the processing of the eye region instead of the processing of the face region has less computational complexity. In the other category, the symptoms of fatigue and distraction are detected not only from eyes, but also from other regions of the face and head. In this approach, in addition to processing of eye region, other symptoms including yawning and head nodding are also extracted.

2.1 EXISTING SYSTEM

In [1] 2019, A.Subbarao, K.Sahithya explained what are the different hardware approaches for driver drowsiness and one of the component they used is the eye blink sensor. A eye blink sensor is used to detect the driver drowsiness and alerts the driver with buzzer.

In [2] 2019, Muhammad Tayab Khan and Hafeez Anwar implemented driver drowsiness detection in real time surveillance videos based on Eyelid Closure The Evalutaion of this system is used three image dataset where images in first data set have a uniform background and second data set is based on face deformations and the third data set of videos of people driving the cars was also used and showed its feasibility for use in real time scenarios.

In [3] 2019, Sukrit Mehta, Sharad Dadhich and Sahil Gumber has used the real time drowsiness detection system using eye aspect ratio and eye closure ratio EAR AND ECR capable of detecting facial landmarks computes ear and ecr to detect based on adaptive thresholding and random forest.

In [4] 2018, Gianfranco Fancello and Mariangela Daga Has researched and performed analysis on driving behaviour for professional bus drivers . The analysis of the data performed using Multiple Correspondences Analysis (MCA) shows that a correlation does exist between the perceived level of discomfort in a set of body areas and the age and body weight of the driver.

In [5] 2017, Ramalatha Marimuthu worked on driver fatigue detection using image processing and accident prevention is carried out by means of image processing and alert system to alert driver by lightling of that particular vehicles lights on and off.

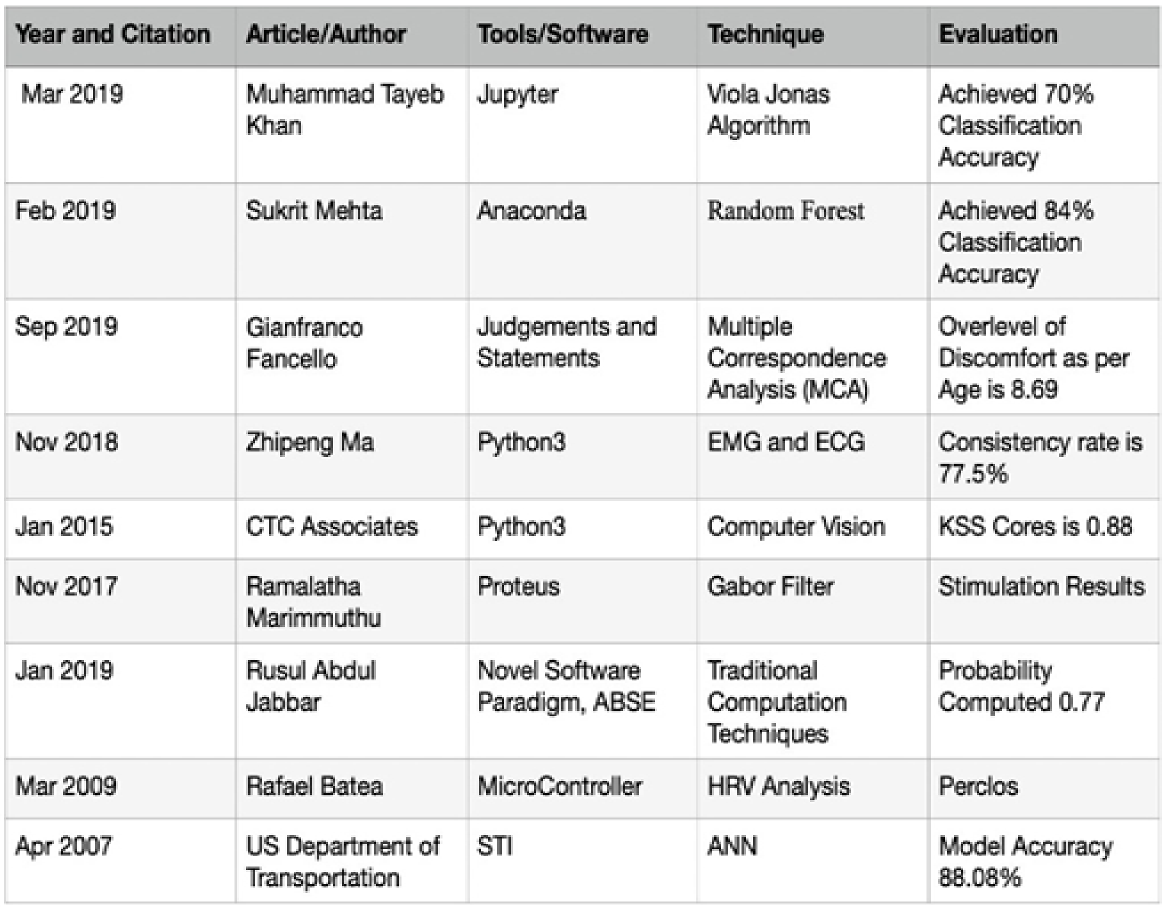
In [6] 2016, Suhas Katkar developed a system which uses IR sensor to sense fatigue and alcohol sensor for drunkenness and provides alarm, stops engine and sends message with location to owner.

In [7] 2015, Raees Ahmad edeveloped a system using a web camera to record the head movements of the driver to detect drowsiness and alert the driver using a signal.

In [8] 2012, Susanna Leanderson Olsson simulated blink behavior based indicators and a lane position monitoring system to alert the owners using vibrations in the steering wheel.

In [9] 2012, uses remotely located charge coupled-device cameras for acquiring the video images of the driver’s face and for illumination uses active infrared lights. Different visual cues like the movements of the eyelid, gaze and head along with facial expression are used to determine fatigue condition.

In [10] 2010,Xiao Fen used a Gabor based representation with dynamic facial image sequences for detecting and monitoring the human fatigue. It uses feature extraction, fusion and Adaboost algorithm to select the most discriminative feature to identify fatigue.



Year and Citation	Article/Author	Tools/Software	Technique	Evaluation
Mar 2019	Muhammad Tayeb Khan	Jupyter	Viola Jonas Algorithm	Achieved 70% Classification Accuracy
Feb 2019	Sukrit Mehta	Anaconda	Random Forest	Achieved 84% Classification Accuracy
Sep 2019	Gianfranco Fancello	Judgements and Statements	Multiple Correspondence Analysis (MCA)	Overlevel of Discomfort as per Age is 8.69
Nov 2018	Zhipeng Ma	Python3	EMG and ECG	Consistency rate is 77.5%
Jan 2015	CTC Associates	Python3	Computer Vision	KSS Cores is 0.88
Nov 2017	Ramalatha Marimmuthu	Proteus	Gabor Filter	Stimulation Results
Jan 2019	Rusul Abdul Jabbar	Novel Software Paradigm, ABSE	Traditional Computation Techniques	Probability Computed 0.77
Mar 2009	Rafael Batea	MicroController	HRV Analysis	Perclos
Apr 2007	US Department of Transportation	STI	ANN	Model Accuracy 88.08%

Fig 1: Research Articles

2.2 PROPOSED SYSTEM

Here in this study, we are trying to show how driver fatigue system is working using different Machine Learning Models for classifying the extracted data. Hence, this drives us to create and analyze the models on this data. Eye aspect ratio from driver’s face are taken out by manually by using pre defined EAR dataset. Collected dataset is applied using many ML models techniques like Random Forest, Logistic regression, Bernoulli Naive Bayes, Support vector machine , KNN etc. Lastly, observing results we can conclude how the models are working for the study and which is the best model for Driver fatigue system. This helps us to know the trends on some models and how people are responding for it.

Chapter – 3

DESIGN FLOW & PROCESS

3.1 PROBLEM STATEMENT

In this project, we try to implement different types of classifiers that helps to overcome the challenges of driver fatigue. Design and implement a Driver Fatigue Monitoring System that utilizes technology to detect and alert drive. Driver fatigue poses a significant challenge in the realm of AI-driven transportation systems, necessitating a comprehensive understanding of the relationship between eye blink patterns and fatigue levels. Despite the availability of datasets capturing various instances of driver fatigue, the correlation between the distance between the two eyelids and the onset of fatigue remains inadequately explored. Therefore, the development of a robust AI model that can accurately analyze and interpret the data pertaining to the distance between the two eyelids in conjunction with established fatigue patterns is imperative. Such a model would significantly contribute to the enhancement of early detection systems for driver fatigue, leading to improved road safety and the prevention of potential accidents.

3.2THEORY

What Is Artificial Intelligence?

Artificial Intelligence is a method of making a computer, a computer-controlled robot, or a software think intelligently like the human mind. AI is accomplished by studying the patterns of the human brain and by analyzing the cognitive process. The outcome of these studies develops intelligent software and systems.

AI systems work by merging large with intelligent, iterative processing algorithms. This combination allows AI to learn from patterns and features in the analyzed data. Each time an Artificial Intelligence system performs a round of data processing, it tests and measures its performance and uses the results to develop additional expertise.

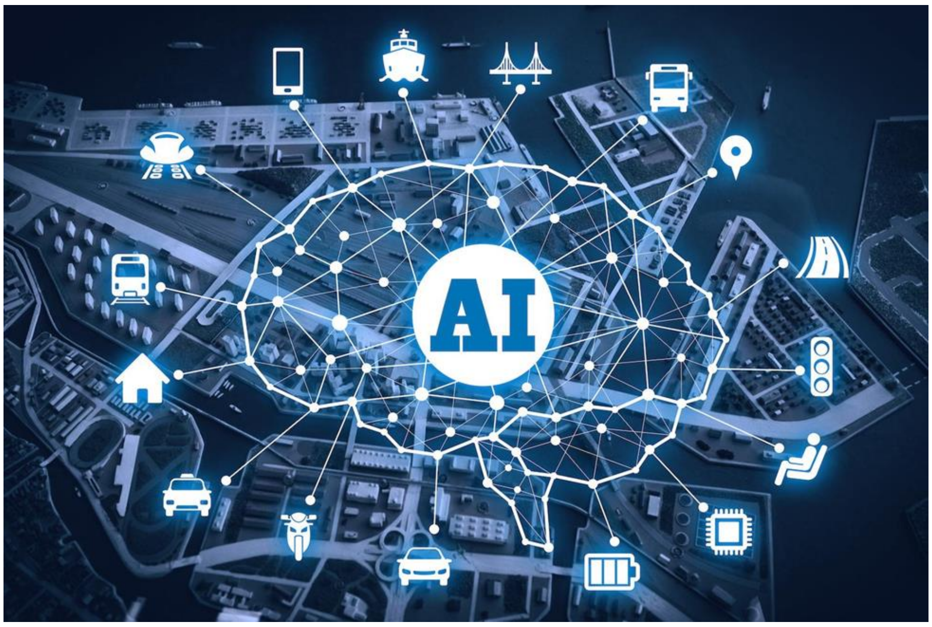


Fig 2 : AI applications

AI in Transport Industry

Artificial Intelligence (AI) in transportation is revolutionizing the mobility industry, leading to significant improvements in efficiency, safety, and convenience. To achieve this, machine learning is becoming more common in many sectors of transportation and mobility.

Scientists have been researching AI for decades. In the past few years, advancements in machine learning have sped up the adaptation of AI in many areas, including transportation and mobility.

One of the key transportation areas where AI is making an impact is autonomous vehicles. Self-driving cars have the potential to reduce accidents caused by human error and improve overall traffic flow. Many major car manufacturers and technology companies are currently developing autonomous vehicles, with some already testing them on public roads.

In many cases, developers train AI control algorithms to reproduce the behavior of experienced drivers navigating through surrounding traffic. PTV Group, for example, collaborates with AI developers in projects like Co-Exist to ensure that autonomous vehicles behave the same in PTV's traffic simulation products.

Overall, AI is transforming the transportation industry, making it more efficient, safe, and convenient for everyone. While there are still challenges to be overcome, such as the need for more data and the development of robust regulations, the benefits of AI in transportation are clear and will continue to grow in the future.

AI is a powerful tool that can help us to create a better transportation system that is safer, more efficient, and more sustainable. With continued research and development, we can look forward to a future where transportation is more accessible, convenient, and green.

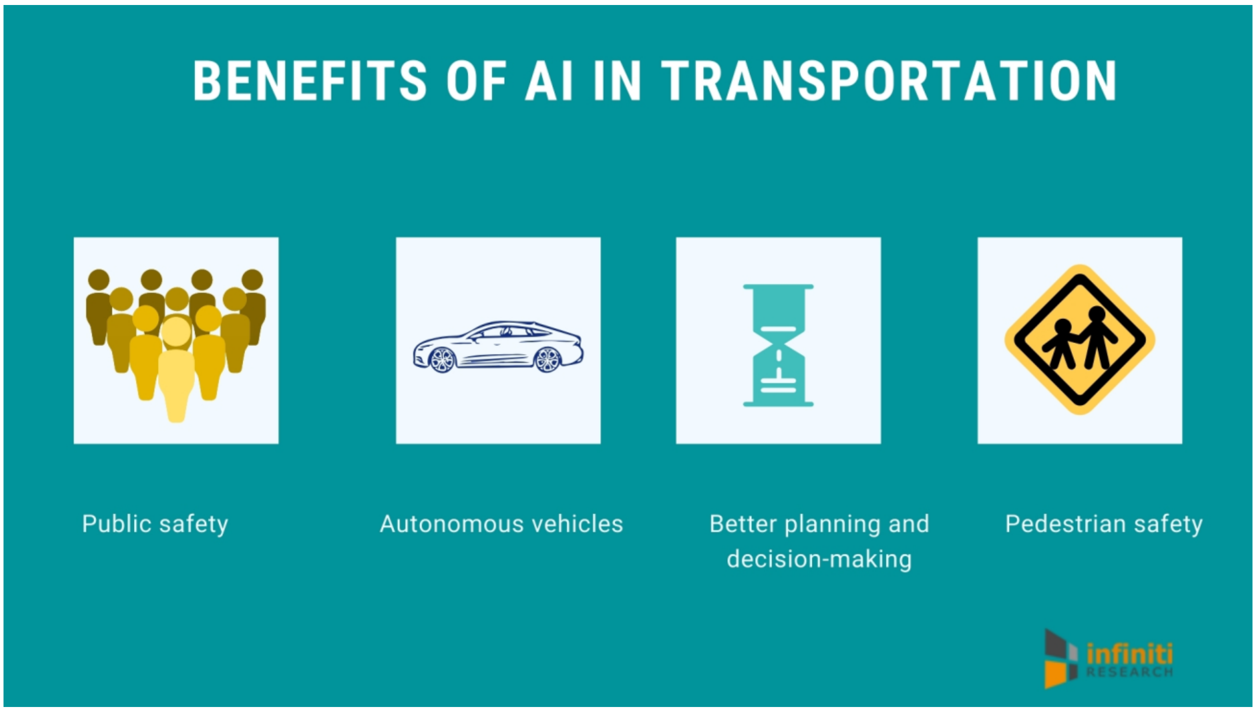

BENEFITS OF AI IN TRANSPORTATION
Public safety
Autonomous vehicles
Better planning and decision-making
Pedestrian safety
infiniti RESEARCH

Fig 3: AI in Transportation

Some key ways AI is transforming the transport sector:

1. Autonomous Vehicles: AI plays a crucial role in the development of self-driving cars and other autonomous vehicles. These vehicles use sensors, cameras, and AI algorithms to perceive their surroundings, make real-time decisions, and navigate without human intervention. Companies like Tesla, Waymo, and General Motors are at the forefront of this technology.

2. Traffic Management: AI is used to optimize traffic flow and reduce congestion. Traffic management systems can analyze real-time data from cameras, sensors, and GPS devices to make dynamic adjustments to traffic signals and routes, reducing traffic

jams and travel times.

3. Predictive Maintenance: AI helps monitor the condition of vehicles and infrastructure, such as trains, buses, and bridges, to predict when maintenance is needed. This proactive approach reduces downtime and improves the safety and reliability of transport systems.

4. Public Transportation Planning: AI-driven algorithms help optimize public transportation routes, schedules, and vehicle allocation, ensuring that services are efficient and meet passenger demand. This can lead to improved public transportation systems that are more convenient and cost-effective.

5. Ride-Sharing and Transportation Apps: Companies like Uber and Lyft use AI to match riders with drivers, estimate arrival times, and calculate fares. These platforms also utilize AI for demand prediction, route optimization, and driver incentives.

6. Supply Chain Management: AI is used in logistics and supply chain management to optimize routes, track shipments, predict delivery times, and manage inventory efficiently. This can lead to cost savings and reduced environmental impact.

7. Safety and Security: AI can enhance safety through features like collision avoidance systems in vehicles, predictive maintenance to prevent accidents, and video analytics for monitoring public transportation and traffic.

8. Personalized Travel Services: AI-powered travel platforms can offer personalized recommendations for travelers, including route suggestions, accommodations, and activities based on individual preferences and past behavior.

9. Environmental Impact: AI can help reduce the environmental impact of transportation by optimizing routes, reducing emissions, and encouraging the use of electric vehicles and other sustainable transportation options.

10. Passenger Experience: AI can enhance the passenger experience through chatbots, voice assistants, and in-vehicle entertainment systems, making travel more comfortable and convenient.

Fig 4: Transport Modes of AI

How does AI help to maintain safety in the transport industry?

Artificial Intelligence (AI) plays a crucial role in enhancing safety in transportation by providing various tools and technologies that help mitigate risks and improve overall safety. Here are several ways AI contributes to safety in transportation:

1. Collision Avoidance Systems: AI-powered sensors and cameras in vehicles can detect potential collisions and provide real-time warnings to drivers or, in the case of autonomous vehicles, take autonomous action to avoid accidents. These systems include features like automatic emergency braking and lane-keeping assistance.
2. Predictive Maintenance: AI is used to monitor the condition of vehicles and infrastructure, predicting when maintenance is required. By identifying and addressing potential issues before they lead to breakdowns, AI reduces the risk of accidents caused by mechanical failures.
3. Driver Assistance Systems: AI-equipped vehicles often feature advanced

driver assistance systems (ADAS) that help drivers navigate safely. These systems include adaptive cruise control, blind-spot monitoring, and automated parking, which reduce the risk of accidents.

4. Traffic Management: AI-based traffic management systems optimize traffic flow and adapt traffic signals in real time to reduce congestion and minimize the risk of traffic accidents caused by gridlock or excessive traffic density.
5. Predictive Analytics: AI analyzes historical and real-time data to predict traffic conditions, weather events, and other factors that may impact transportation safety. This information can be used to plan routes and take preventive measures.
6. Emergency Response and Coordination: AI can improve emergency response systems by providing faster and more accurate information about accidents, incidents, and road conditions to first responders. This helps save lives and reduce the severity of accidents.
7. Infrastructure Safety: AI helps monitor the safety of transportation infrastructure, such as bridges and tunnels, by detecting signs of wear, damage, or other issues that could lead to accidents.
8. Monitoring and Surveillance: AI-powered video analytics and surveillance systems are used to monitor traffic and public transportation, enabling authorities to respond to incidents quickly and improve safety.
9. Automated Warning Systems: AI can provide automated warnings and alerts to drivers, pedestrians, and cyclists in real time, helping prevent accidents and increase awareness of potential dangers on the road.
10. Advanced Vehicle Communication: AI facilitates communication between vehicles (vehicle-to-vehicle or V2V) and between vehicles and infrastructure (vehicle-to-infrastructure or V2I), allowing vehicles to share information about their speed, location, and other factors, which can help prevent accidents.
11. Data Analysis and Investigation: After an accident or incident, AI can help authorities analyze data from black boxes, traffic cameras, and other sources to understand the causes and improve safety measures.
12. Driver Behavior Monitoring: AI-powered telematics systems monitor driver behavior, providing feedback to drivers and fleet managers to promote safe driving practices.

Overall, AI contributes to safety in transportation by reducing the likelihood of accidents, improving emergency response, and helping to create safer and more efficient transportation systems. As AI technologies continue to advance, they are expected to play an even more significant role in enhancing transportation safety.

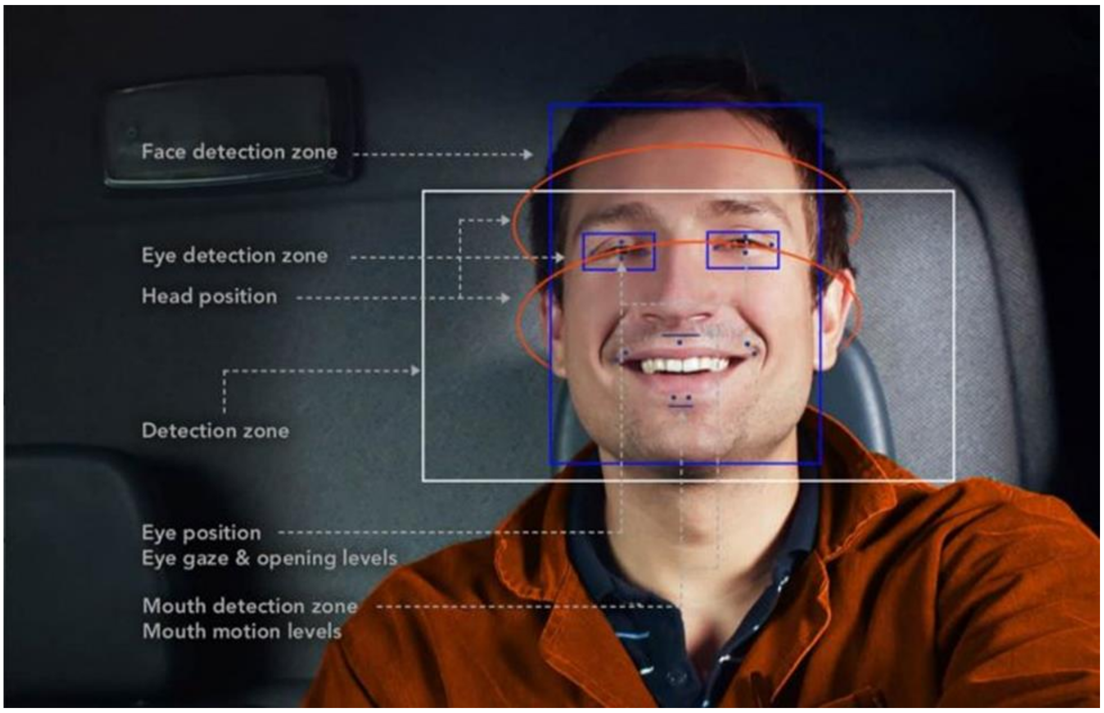


Fig 5: DFMS

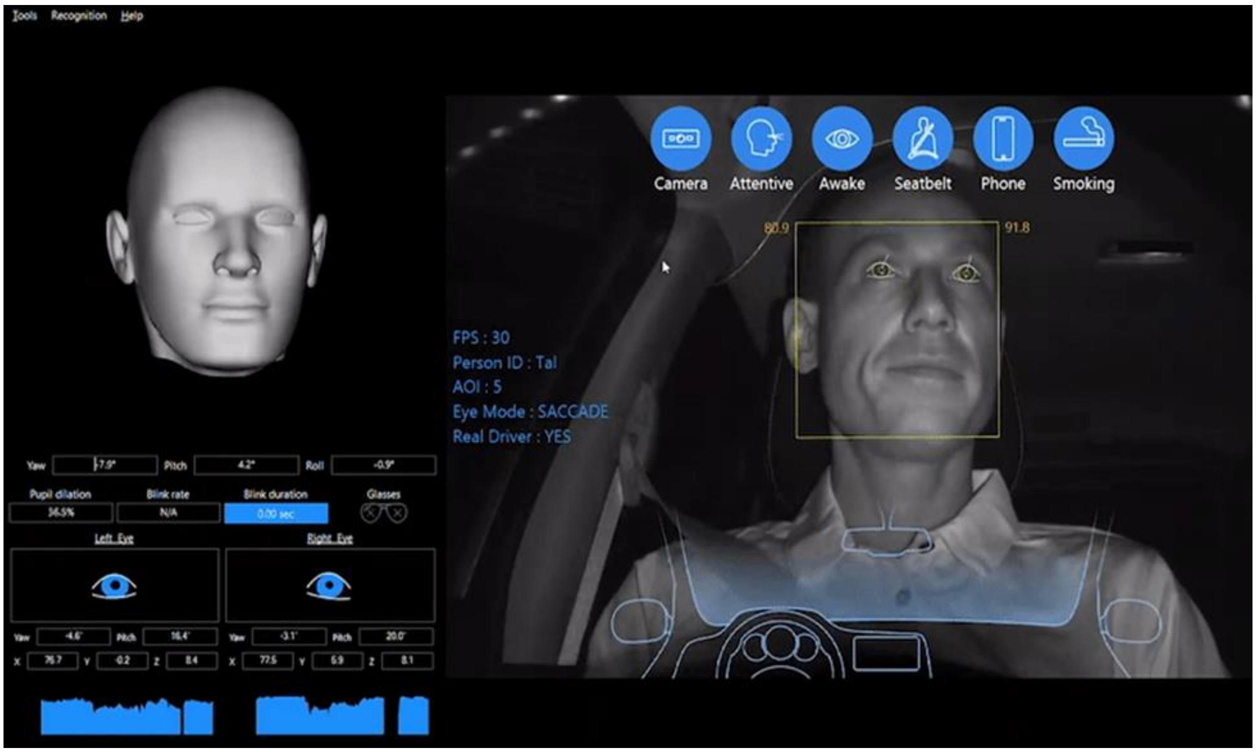
DRIVER DROWSINESS MONITORING

To encourage efficient and safe driving, drivers are screened and evaluated on driving habits. Advanced Driver Drowsiness Monitoring System or Driver Fatigue Monitoring System to detect and monitor behavior and fatigue levels of drivers are emerging which makes the cars more intelligent for avoiding accidents on roads. Systems are being developed for real-time monitoring of vehicles which controls the speed of the vehicle and fatigue level of the driver to prevent accidents. The primary components of such a system will be microcontrollers along with some sensors like an eye blink, gas, impact sensors, alcohol detecting sensor, and fuel sensors. GPS and Google Maps APIs are used to track the location of the vehicle which can be sent to a predefined number in the system.

The Driver Drowsiness Monitoring System uses automotive-grade image sensors that capture infrared images of the driver's eyes, using patented pupil identification technology and a high-speed digital signal processor to analyze & identify if the driver is either drowsy or distracted. The contactless setup and a sophisticated algorithm give the device the ability to understand the state of drivers even in tricky conditions like in presence of strong lighting during noon hours, drivers wearing sunglasses, etc.

In the context of driving, drowsiness detection systems are employed to monitor a driver's alertness and intervene when signs of fatigue are detected. These systems typically utilize various sensors and technologies to assess behavioral and physiological indicators associated with drowsiness. Common methods include:

- Computer Vision: Analyzing facial features and eye movements to detect signs like eyelid drooping, slow blinks, or changes in gaze patterns.
- Physiological Monitoring: Using sensors to measure physiological signals such as heart rate variability, brainwave patterns (Electroencephalography or EEG), or muscle activity to assess the driver's state of alertness.
- Behavioral Analysis: Monitoring driving behavior, such as lane deviations, erratic steering, or sudden corrections, to identify patterns indicative of drowsiness.
- Machine Learning: Employing machine learning algorithms to analyze patterns and trends in data collected from various sensors, training the system to recognize precursors to drowsiness.



FPS : 30
Person ID : Tal
AOI : 5
Eye Mode : SACCADE
Real Driver : YES

Fig 6: EAR points and ratios

Advanced Driving Assistance System, based on AI, Machine Learning & our proprietary neural engine & computer vision technology is capable of reducing collision accidents with an early warning when it detects potential collision dangers during driving, which also helps to improve driving behaviors and overall driving safety.

Facial Recognition:

•Our platform provides facial recognition using IR Cameras, Geometrical Face Mapping, and Portrait Database to ensure a personalized experience for drivers.

•IR Cameras and Geometrical 3D Mapping create a precise image of the person. Neural Networks and Machine Learning are used to improve recognition efficiency and accuracy through image classifiers and a GPU-based neural engine.

•Facial recognition enables custom settings for different drivers of the car. Drivers may have choices of the playlist, predetermined speed limits, automated climate control, etc.

•A driver fatigue detection system is a technology that helps identify and alert drivers when they show signs of fatigue or drowsiness while operating a vehicle. Fatigue is a leading cause of accidents on the road, as it impairs a driver's reaction time, decision-making, and overall alertness. Driver fatigue detection systems aim to enhance road safety by preventing accidents caused by drowsy or inattentive drivers.

•These systems typically use a combination of sensors and artificial intelligence algorithms to monitor various aspects of the driver's behavior and physiological state.

Why perform Drowsiness Detection

Performing drowsiness detection is essential for several reasons, primarily centered around enhancing safety in various activities, with a particular focus on preventing accidents and improving overall well- being. Here are key reasons for implementing drowsiness detection:

Accident Prevention:

Drowsy driving or operating machinery poses a significant risk of accidents. Drowsiness detection systems are crucial in identifying early signs of fatigue, allowing for timely interventions to prevent accidents caused by impaired alertness and delayed reaction times.

Road Safety:

In the context of driving, drowsiness is a major contributing factor to road accidents. Detecting drowsiness helps mitigate the risk of collisions, lane departures, and other incidents by alerting the driver and encouraging them to take necessary breaks or rest.

Occupational Safety:

In occupations where sustained attention is critical, such as pilots, train operators, or heavy machinery operators, drowsiness detection ensures that individuals are alert and capable of performing their tasks safely. This is particularly important in industries where lapses in concentration can have severe consequences.

Individual Well-Being:

Drowsiness detection contributes to the overall health and well-being of individuals by promoting healthy sleep patterns and preventing the negative effects of fatigue. Adequate rest is essential for cognitive function, mood, and overall physical health.

Productivity and Performance:

In work environments where individuals are required to maintain focus and productivity, drowsiness detection systems help identify periods of reduced alertness. Addressing drowsiness can lead to improved productivity, decision-making, and job performance.

Preventing Microsleeps:

Drowsiness detection is crucial in preventing microsleeps, brief episodes of unintended sleep that can occur without an individual's awareness. Microsleeps can be particularly dangerous, especially in situations where sustained attention is critical, such as driving or operating machinery.

Reducing Fatigue-Related Costs:

Fatigue-related accidents and incidents can result in substantial economic costs for individuals and organizations. Drowsiness detection systems help reduce these costs by preventing accidents, injuries, and associated expenses.

Legal Compliance:

In some industries, there are legal regulations and guidelines that mandate the implementation of drowsiness detection systems to ensure compliance with safety standards. Adhering to these regulations is essential for maintaining a safe working environment and avoiding legal implications.

Types of Drowsiness Detection Systems:

Drowsiness Detection Systems (DDS) come in various types, each employing different technologies and methodologies to identify signs of drowsiness. Here are some common types of DDS:

Computer Vision-Based Systems:

These systems use cameras and computer vision algorithms to analyze facial expressions, eye movements, and other visual cues indicative of drowsiness. Facial recognition and eye-tracking technologies play a key role in identifying signs such as eyelid drooping, slow blinks, or changes in gaze patterns.

Physiological Monitoring Systems:

These systems monitor physiological signals to assess the driver’s state of alertness. Common physiological measures include Electroencephalography (EEG) to track brainwave patterns, Electrocardiography (ECG) for heart rate variability, and other sensors to measure muscle activity. Changes in these signals can indicate drowsiness.

Behavioral Analysis Systems:

These systems focus on analyzing the driver's behavior, such as steering patterns, lane deviations, and changes in driving style. Anomalies or patterns associated with drowsiness trigger alerts or interventions.

Machine Learning-Driven Systems:

Machine learning algorithms, including supervised and unsupervised learning models, are employed to analyze patterns in driving behavior and physiological signals. These systems can adapt and learn from data, improving accuracy over time.

Multimodal Integration Systems:

Combining multiple modalities, such as computer vision, physiological monitoring, and machine learning, these systems aim to create a more comprehensive and accurate drowsiness detection solution. The integration of diverse data sources enhances adaptability and robustness.

In-Car Monitoring Systems:

These systems are integrated directly into the vehicle and can include features such as steering wheel sensors, seat belt sensors, or infrared sensors to monitor the driver’s movements and physiological signals. They often provide real-time feedback or alerts.

Wearable Devices:

Wearable technologies, such as smartwatches or headsets, can incorporate sensors to monitor physiological signals and detect drowsiness. These devices may provide haptic feedback or alerts to the wearer.

Smartphone Applications:

Some drowsiness detection systems leverage smartphone sensors, such as accelerometers and gyroscopes, to analyze the driver’s movements and behavior. These applications can issue alerts or warnings to the driver.

Lane Departure Warning Systems (LDWS):

While not exclusively focused on drowsiness, LDWS are part of some DDS. These systems use cameras to monitor lane position and issue warnings if the vehicle deviates from its lane, which can be indicative of drowsiness or distraction.

Biometric Wearables for Fatigue Monitoring:

Specialized biometric wearables focus on monitoring fatigue-related indicators, including heart rate variability and skin conductance. These wearables are designed for continuous fatigue monitoring in various settings.

How does Drowsiness Detection Systems work?

Drowsiness Detection Systems (DDS) employ various technologies and methodologies to monitor and identify signs of drowsiness in individuals. The specific workings of a DDS can vary based on the type of system and the technologies it incorporates. Here's a general overview of how DDS typically work:

Data Acquisition: Sensors and Data Sources: DDS rely on different sensors and data sources, depending on their type. These can include cameras for computer vision, physiological sensors (such as EEG or ECG), steering sensors, accelerometers, and other relevant data sources.

Data Processing: Computer Vision Algorithms: In systems using computer vision, facial recognition algorithms and image processing techniques analyze visual data to detect facial features and movements. This includes monitoring eye closure, blink rate, head position, and facial expressions associated with drowsiness. Physiological Signal Analysis: Systems that monitor physiological signals process data from sensors like EEG or ECG to identify patterns indicative of drowsiness. Changes in brainwave activity, heart rate variability, or muscle activity can be key indicators.

Feature Extraction: Identifying Key Features: Machine learning-driven DDS extract relevant features from the data. For example, in computer vision-based systems, features might include the frequency of eye blinks or the duration of eyelid closure. In physiological monitoring, features could be derived from specific characteristics of EEG or ECG signals.

Model Training (for Machine Learning Systems): Supervised Learning: Machine learning models are trained on labeled datasets that include examples of drowsy and non-drowsy states. The models learn to recognize patterns associated with drowsiness during this training phase.

Unsupervised Learning: In some cases, unsupervised learning models analyze data without predefined labels, identifying patterns or anomalies that indicate drowsiness.

Real-Time Analysis: Continuous Monitoring: DDS continuously monitor the relevant signals and data streams in real-time during operation. This allows for immediate detection of drowsiness-related patterns or changes in behavior.

Decision-Making and Alerting: Thresholds and Rules: The system sets predefined thresholds or rules based on the learned patterns. When the observed features surpass these thresholds, the system determines that drowsiness is likely. Alerts and Interventions: Upon detecting signs of drowsiness, the DDS issues alerts to the individual. These alerts can take various forms, such as audible alarms, visual cues, haptic feedback, or other interventions designed to prompt the individual to take corrective action.

Adaptability (for Some Systems): Machine Learning Adaptation: Machine learning-driven DDS may continuously adapt and refine their models over time based on ongoing data. This adaptability helps improve accuracy and accommodate variations in individual behavior.

Integration with Vehicle Systems (for In-Car Systems): Automatic Vehicle Interventions: Some DDS integrated into vehicles may have the capability to trigger automatic interventions, such as adjusting the seat position, activating vibration alerts, or modifying vehicle settings to ensure the driver's attention is regained.

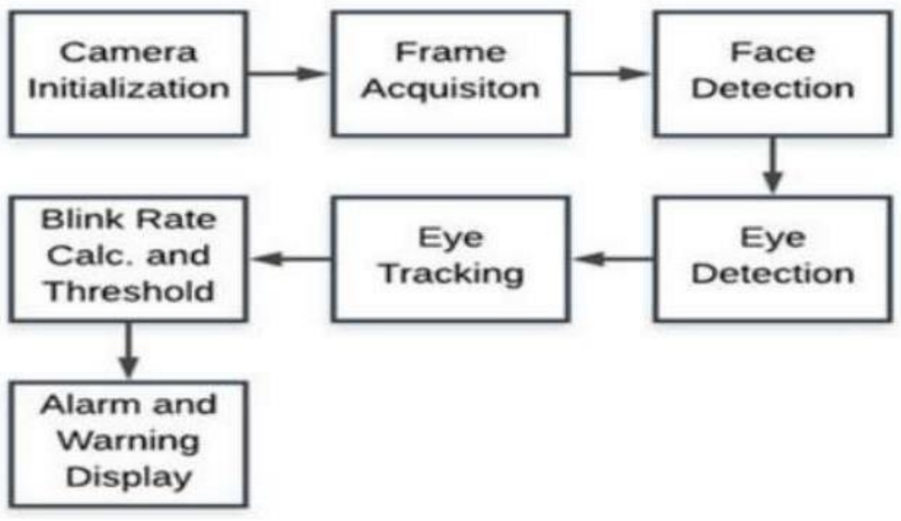


Fig7: Stepwise process of DDS

Challenges of Drowsiness Detection System

The development and deployment of Drowsiness Detection Systems (DDS) come with various challenges that researchers and engineers must address to ensure the effectiveness and reliability of these systems. Some of the key challenges include:

Individual Variability:

People exhibit diverse patterns of drowsiness, making it challenging to create a one-size-fits-all detection model. Variability in physiological responses, facial expressions, and driving behaviors poses a significant challenge in designing systems that can adapt to individual differences.

Environmental Factors:

Driving conditions and environments can vary widely, introducing challenges for DDS. Factors such as changes in lighting, road conditions, and weather can affect the accuracy and reliability of detection systems.

Real-Time Implementation:

Achieving real-time processing and response is crucial for effective drowsiness detection. Delays in alerting the driver or triggering interventions can compromise the system's ability to prevent accidents. Overcoming computational and processing constraints is essential.

False Positives and False Negatives:

Balancing the detection of genuine instances of drowsiness while minimizing false alarms (false positives) and ensuring that actual drowsy states are not overlooked (false negatives) is a delicate challenge. Striking the right balance is essential for user acceptance and system effectiveness.

Ethical Considerations and Privacy:

Collecting and processing personal data, especially physiological signals and facial images, raises ethical concerns. Ensuring user privacy, obtaining informed consent, and implementing secure data storage and transmission are critical considerations for DDS developers.

User Acceptance and Intrusiveness:

DDS need to be accepted and adopted by users for them to be effective. Intrusive sensors or constant alerts may lead to user discomfort or resistance. Striking a balance between system effectiveness and user acceptance is crucial.

Adaptability to Driving Styles:

Drivers exhibit a wide range of driving styles, and DDS must adapt to these variations. Developing systems that can effectively identify drowsiness across different driving behaviors and preferences is a significant challenge.

Validation in Real-World Scenarios:

Conducting thorough real-world validations is essential to ensure that DDS perform reliably in diverse driving conditions. Simulations and controlled experiments may not capture the complexity of actual driving scenarios, making real-world testing crucial but challenging.

Interference and Noise:

External factors, such as noise, vibrations, or interference from other electronic devices in the vehicle, can impact the accuracy of sensors and data streams. DDS must be robust enough to filter out irrelevant signals and focus on indicators of drowsiness.

Cross-Cultural Considerations:

DDS effectiveness may be influenced by cultural differences in facial expressions and driving behaviors. Ensuring that the system works well across diverse cultural contexts is an additional challenge for developers.

What is Machine Learning?

Machine Learning is said as a subset of artificial intelligence that is mainly concerned with the development of algorithms which allow a computer to learn from the data and past experiences on their own. The term machine learning was first introduced by Arthur Samuel in 1959.

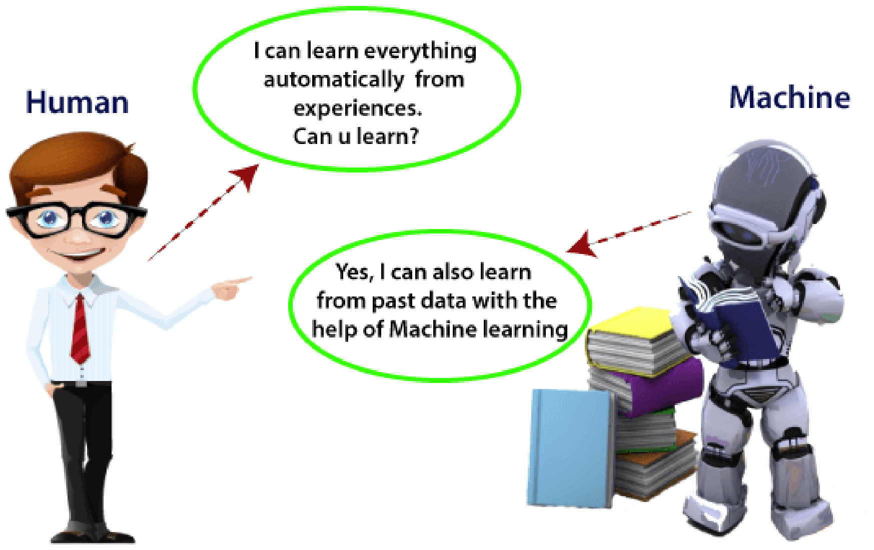

Human
I can learn everything automatically from experiences. Can u learn?
Machine
Yes, I can also learn from past data with the help of Machine learning

Fig8: Way of Learning among Machines and Humans

Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed.

How does Machine Learning work:

A Machine Learning system learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it. The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately.

Suppose we have a complex problem, where we need to perform some predictions, so instead of writing a code for it, we just need to feed the data to generic

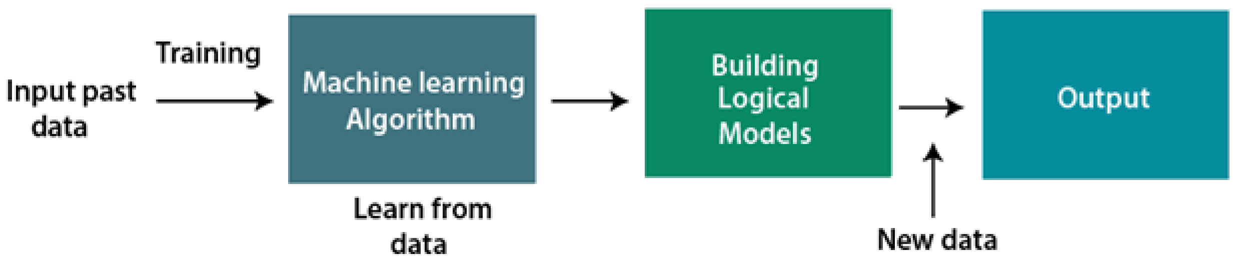


Fig9: Above, the machine learning process

algorithms, and with the help of these algorithms, machine builds the logic as per the data and predict the output. Machine learning has changed our way of thinking about the problem.

The below block diagram explains the working of Machine Learning algorithm:

Features of Machine Learning:
• Machine learning uses data to detect various patterns in a given dataset.
• It can learn from past data and improve automatically.
• It is a data-driven technology.
• Machine learning is much similar to data mining as it also deals with the huge amount of the data.

What is Deep Learning?

Deep learning is based on the branch of machine learning, which is a subset of artificial intelligence. Since neural networks imitate the human brain and so deep learning will do. In deep learning, nothing is programmed explicitly. Basically, it is a machine learning class that makes use of numerous nonlinear processing units so as to perform feature extraction as well as transformation. The output from each preceding layer is taken as input by each one of the successive layers.

Deep learning models are capable enough to focus on the accurate features themselves by requiring a little guidance from the programmer and are very helpful in solving out the problem of dimensionality. Deep learning algorithms are used, especially when we have a huge no of inputs and outputs.

Since deep learning has been evolved by the machine learning, which itself is a subset of artificial intelligence and as the idea behind the artificial intelligence is to mimic the human behavior, so same is "the idea of deep learning to build such algorithm that can mimic the brain".

Deep learning is implemented with the help of Neural Networks, and the idea behind the motivation of Neural Network is the biological neurons, which is nothing but a brain cell.

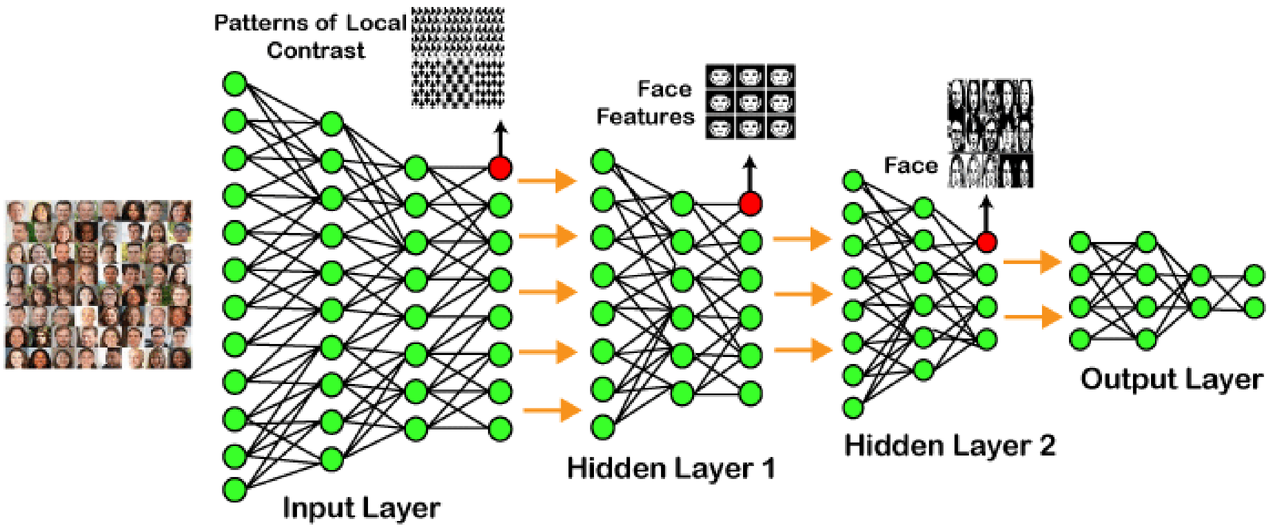


Fig10. Example of Deep Learning

In the example given above, we provide the raw data of images to the first layer of the input layer. After then, these input layer will determine the patterns of local contrast that means it will differentiate on the basis of colors, luminosity, etc. Then the 1st hidden layer will determine the face feature, i.e., it will fixate on eyes, nose, and lips, etc. And then, it will fixate those face features on the correct face template. So, in the 2nd hidden layer, it will actually determine the correct face here as it can be seen in the above image, after which it will be sent to the output layer. Likewise, more hidden layers can be added to solve more complex problems, for example, if you want to find out a particular kind of face having large or light complexions. So, as and when the hidden layers increase, we are able to solve complex problems.

Architectures

Deep Neural Networks

It is a neural network that incorporates the complexity of a certain level, which means several numbers of hidden layers are encompassed in between the input and output layers. They are highly proficient on model and process non-linear associations.

Deep Belief Networks

A deep belief network is a class of Deep Neural Network that comprises of multi-layer belief networks. Steps to perform DBN:

With the help of the Contrastive Divergence algorithm, a layer of features is learned from perceptible units.

Next, the formerly trained features are treated as visible units, which perform learning of features. Lastly, when the learning of the final hidden layer is accomplished, then the whole DBN is trained.

Recurrent Neural Networks
It permits parallel as well as sequential computation, and it is exactly similar to that of the human brain (large feedback network of connected neurons). Since they are capable enough to reminisce all of the imperative things related to the input they have received, so they are more precise.

Types of Deep Learning Networks

1. Feed Forward Neural Network
A feed-forward neural network is none other than an Artificial Neural Network, which ensures that the nodes do not form a cycle. In this kind of neural network, all the perceptron are organized within layers, such that the input layer takes the input, and the output layer generates the output. Since the hidden layers do not link with the outside world, it is named as hidden layers. Each of the perceptron contained in one single layer is associated with each node in the subsequent layer. It can be concluded that all of the nodes are fully connected. It does not contain any visible or invisible connection between the nodes in the same layer. There are no back-loops in the feed-forward network. To minimize the prediction error, the backpropagation algorithm can be used to update the weight values.

Applications:
• Data Compression
• Pattern Recognition
• Computer Vision
• Sonar Target Recognition
• Speech Recognition
• Handwritten Characters Recognition

2. Recurrent Neural Network
Recurrent neural networks are yet another variation of feed-forward networks. Here each of the neurons present in the hidden layers receives an input with a specific delay in time. The Recurrent neural network mainly accesses the preceding info of existing iterations. For example, to guess the succeeding word in any sentence, one

must have knowledge about the words that were previously used. It not only processes the inputs but also shares the length as well as weights crossways time. It does not let the size of the model to increase with the increase in the input size. However, the only problem with this recurrent neural network is that it has slow computational speed as well as it does not contemplate any future input for the current state. It has a problem with reminiscing prior information.

Applications:
• Machine Translation
• Robot Control
• Time Series Prediction
• Speech Recognition
• Speech Synthesis
• Time Series Anomaly Detection
• Rhythm Learning
• Music Composition

3. Convolutional Neural Network
Convolutional Neural Networks are a special kind of neural network mainly used for image classification, clustering of images and object recognition. DNNs enable unsupervised construction of hierarchical image representations. To achieve the best accuracy, deep convolutional neural networks are preferred more than any other neural network.

Applications:
• Identify Faces, Street Signs, Tumors.
• Image Recognition.
• Video Analysis.
• NLP.
• Anomaly Detection.
• Drug Discovery.
• Checkers Game.
• Time Series Forecasting.

4. Restricted Boltzmann Machine
RBMs are yet another variant of Boltzmann Machines. Here the neurons present in the input layer and the hidden layer encompasses symmetric connections amid them. However, there is no internal association within the respective layer. But in contrast to RBM, Boltzmann machines do encompass internal connections inside the hidden layer. These restrictions in BMs helps the model to train efficiently.

Applications:
• Filtering.
• Feature Learning.
• Classification.
• Risk Detection.
• Business and Economic analysis.

5. Autoencoders
An autoencoder neural network is another kind of unsupervised machine learning algorithm. Here the number of hidden cells is merely small than that of the input cells. But the number of input cells is equivalent to the number of output cells. An autoencoder network is trained to display the output similar to the fed input to force AEs to find common patterns and generalize the data. The autoencoders are mainly used for the smaller representation of the input. It helps in the reconstruction of the original data from compressed data. This algorithm is comparatively simple as it only necessitates the output identical to the input.
Encoder: Convert input data in lower dimensions. Decoder: Reconstruct the compressed data.

Applications:
• Classification.
• Clustering.
• Feature Compression.

Deep Learning Applications

Self-Driving Cars
In self-driven cars, it is able to capture the images around it by processing a huge amount of data, and then it will decide which actions should be incorporated to take a left or right or should it stop. So, accordingly, it will decide what actions it should take, which will further reduce the accidents that happen every year.

Voice Controlled Assistance
When we talk about voice control assistance, then Siri is the one thing that comes into our mind. So, you can tell Siri whatever you want it to do it for you, and it will search it for you and display it for you.

Automatic Image Caption Generation
Whatever image that you upload, the algorithm will work in such a way that it will generate caption accordingly. If you say blue colored eye, it will display a blue-colored eye with a caption at the bottom of the image.

Automatic Machine Translation
With the help of automatic machine translation, we are able to convert one language into another with the help of deep learning.

Limitations:
• It only learns through the observations.
• It comprises of biases issues.

Advantages:
• It lessens the need for feature engineering.
• It eradicates all those costs that are needless.
• It easily identifies difficult defects.
• It results in the best-in-class performance on problems
.
Disadvantages:
• It requires an ample amount of data.
• It is quite expensive to train.
• It does not have strong theoretical groundwork.

What is Data Preprocessing?

Data preprocessing refers to the set of techniques and operations applied to raw data before it is used for analysis or modeling. It involves transforming, cleaning, and organizing the data to ensure its quality, consistency, and suitability for further processing. Data preprocessing plays a crucial role in data analysis and machine learning tasks, as it helps improve the accuracy, reliability, and efficiency of subsequent data processing steps.

The main steps involved in data preprocessing are as follows:

Data Cleaning:
- Handling missing data: Dealing with missing values by either imputing them or removing rows or columns with missing data.
- Removing duplicates: Identifying and eliminating duplicate records or instances from the dataset. Handling outliers: Detecting and handling outliers or anomalies that may significantly affect the analysis or modeling results.

Data Transformation:
- Feature scaling: Normalizing or standardizing numeric features to bring them to a similar scale and prevent biases in certain algorithms.

- Feature encoding: Converting categorical variables into numerical representations that machine learning algorithms can process.
- Feature discretization: Grouping continuous variables into discrete bins or intervals to simplify the data representation.
- Feature engineering: Creating new features or transforming existing features to better represent the underlying patterns or relationships in the data.

Data Integration:

- Combining data from multiple sources or different datasets into a unified format for analysis or modeling.
- Resolving data inconsistencies, such as conflicting attribute names or data formats, during the integration process.

Data Reduction:

- Dimensionality reduction: Reducing the number of features or variables while preserving the most important information, typically achieved through techniques like Principal Component Analysis (PCA) or feature selection algorithms.
- Instance sampling: Selecting a representative subset of instances or records from a large dataset to reduce computational complexity or balance class distributions.

Data Formatting:

- Ensuring the data is in the appropriate format and structure for analysis or modeling tasks.
- Handling date and time formats, converting text to lowercase or uppercase, or adjusting data representations to match specific requirements.

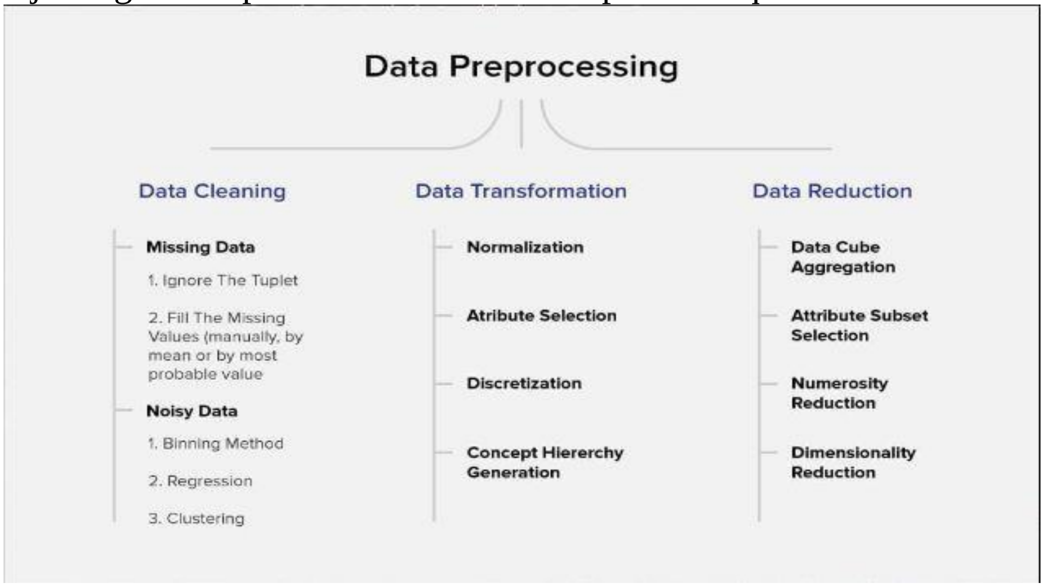


Fig11. Steps of Data Pre-processing

What is Deep Learning Algorithm?

Deep learning can be defined as the method of machine learning and artificial intelligence that is intended to intimidate humans and their actions based on certain human brain functions to make effective decisions. It is a very important data science element that channels its modeling based on data-driven techniques under predictive modeling and statistics. To drive such a human-like ability to adapt and learn and to function accordingly, there have to be some strong forces which we popularly called algorithms.

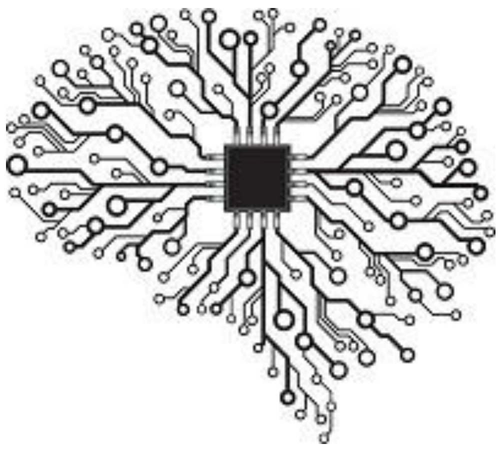


Fig12: Artificial Intelligence as a Human Brain

Deep learning algorithms are dynamically made to run through several layers of neural networks, which are nothing but a set of decision-making networks that are pre-trained to serve a task. Later, each of these is passed through simple layered representations and move on to the next layer. However, most machine learning is trained to work fairly well on datasets that have to deal with hundreds of features or columns. For a data set to be structured or unstructured, machine learning tends to fail mostly because they fail to recognize a simple image having a dimension of 800x1000 in RGB. It becomes quite unfeasible for a traditional machine learning algorithm to handle such depths. This is where deep learning.

Importance of Deep Learning

Deep learning algorithms play a crucial role in determining the features and can handle the large number of processes for the data that might be structured or unstructured. Although, deep learning algorithms can overkill some tasks that might involve complex problems because they need access to huge amounts of data so that they can function effectively. For example, there's a popular deep learning tool that recognizes images namely Imagenet that has access to 14 million images in its dataset-driven algorithms. It is a highly comprehensive tool that has defined a next-level benchmark for deep learning tools that aim images as their dataset.

Deep learning algorithms are highly progressive algorithms that learn about the image that we discussed previously by passing it through each neural network layer. The layers are highly sensitive to detect low-level features of the image like edges and pixels and henceforth the combined layers take this information and form holistic representations by comparing it with previous data. For example, the middle layer might be programmed to detect some special parts of the object in the photograph which other deep trained layers are programmed to detect special objects like dogs, trees, utensils, etc.

However, if we talk out the simple task that involves less complexity and a data-driven resource, deep learning algorithms fail to generalize simple data. This is one of the main reasons deep learning is not considered effective as linear or boosted tree models. Simple models aim to churn out custom data, track fraudulent transactions and deal with less complex datasets with fewer features. Also, there are various cases like multiclass classification where deep learning can be effective because it involves smaller but more structured datasets but is not preferred usually.

Having said that, let's look understand some of the most important deep learning algorithms given below.

Deep Learning Algorithms

The Deep Learning Algorithms are as follows:

1. Convolutional Neural Networks (CNNs)

CNN's popularly known as ConvNets majorly consists of several layers and are specifically used for image processing and detection of objects. It was developed in 1998 by Yann LeCun and was first called LeNet. Back then, it was developed to recognize digits and zip code characters. CNNs have wide usage in identifying the image of the satellites, medical image processing, series forecasting, and anomaly detection.

CNNs process the data by passing it through multiple layers and extracting features to exhibit convolutional operations. The Convolutional Layer consists of Rectified Linear Unit (ReLU) that outlasts to rectify the feature map. The Pooling layer is used to rectify these feature maps into the next feed. Pooling is generally a sampling algorithm that is down-sampled and it reduces the dimensions of the feature map. Later, the result generated consists of 2-D arrays consisting of single, long, continuous, and linear vector flattened in the map. The next layer i.e., called Fully Connected Layer which forms the flattened matrix or 2-D array fetched from the Pooling Layer as input and identifies the image by classifying it.

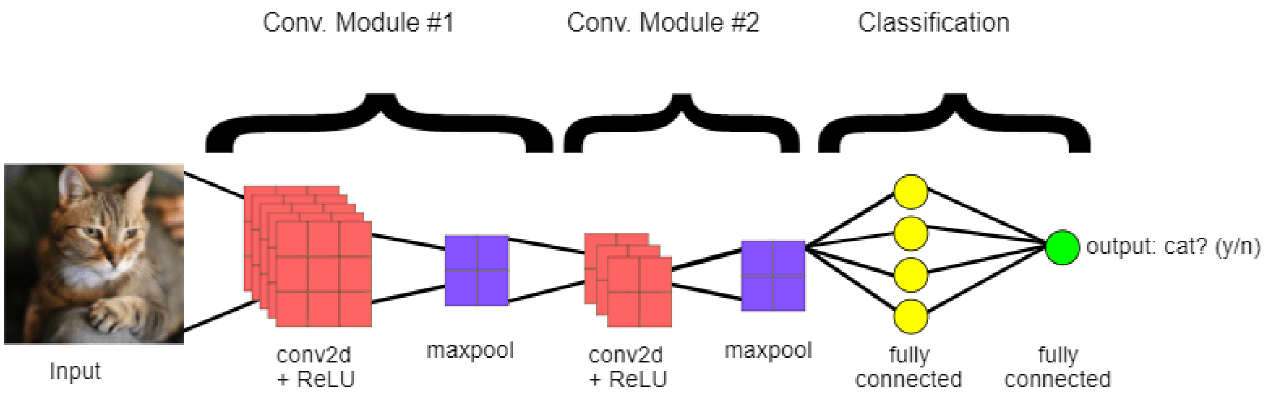


Fig13: Layers in CNN

2. Long Short Term Memory Networks (LSTMs)

LSTMs can be defined as Recurrent Neural Networks (RNN) that are programmed to learn and adapt for dependencies for the long term. It can memorize and recall past data for a greater period and by default, it is its sole behavior. LSTMs are designed to retain over time and henceforth they are majorly used in time series predictions because they can restrain memory or previous inputs. This analogy comes from their chain-like structure consisting of four interacting layers that communicate with each other differently. Besides applications of time series prediction, they can be used to construct speech recognizers, development in pharmaceuticals, and composition of music loops as well.

LSTM work in a sequence of events. First, they don't tend to remember irrelevant details attained in the previous state. Next, they update certain cell-state values selectively and finally generate certain parts of the cell-state as output. Below is the diagram of their operation.

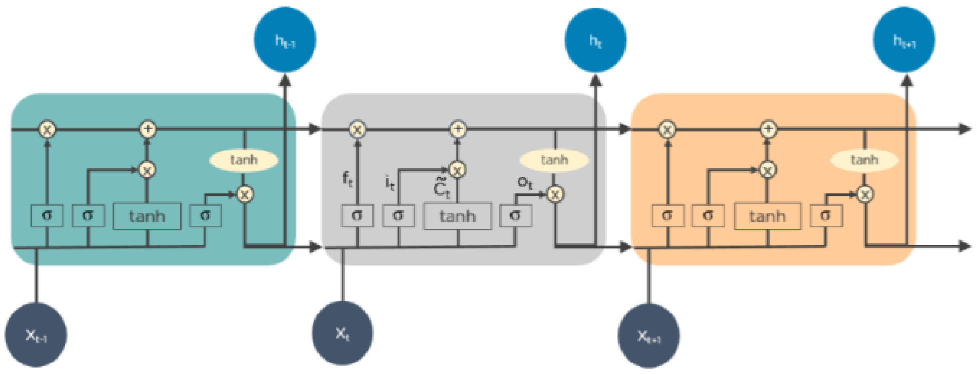


Fig14: Architecture of LSTM

3. Recurrent Neural Networks (RNNs)

Recurrent Neural Networks or RNNs consist of some directed connections that form a cycle that allow the input provided from the LSTMs to be used as input in the current phase of RNNs. These inputs are deeply embedded as inputs and enforce the memorization ability of LSTMs lets these inputs get absorbed for a period in the internal memory. RNNs are therefore dependent on the inputs that are preserved by LSTMs and work under the synchronization phenomenon of LSTMs. RNNs are mostly used in captioning the image, time series analysis, recognizing handwritten data, and translating data to machines.

RNNs follow the work approach by putting output feeds (t-1) time if the time is defined as t. Next, the output determined by t is feed at input time t+1. Similarly, these processes are repeated for all the input consisting of any length. There's also a fact about RNNs is that they store historical information and there's no increase in the input size even if the model size is increased. RNNs look something like this when unfolded.

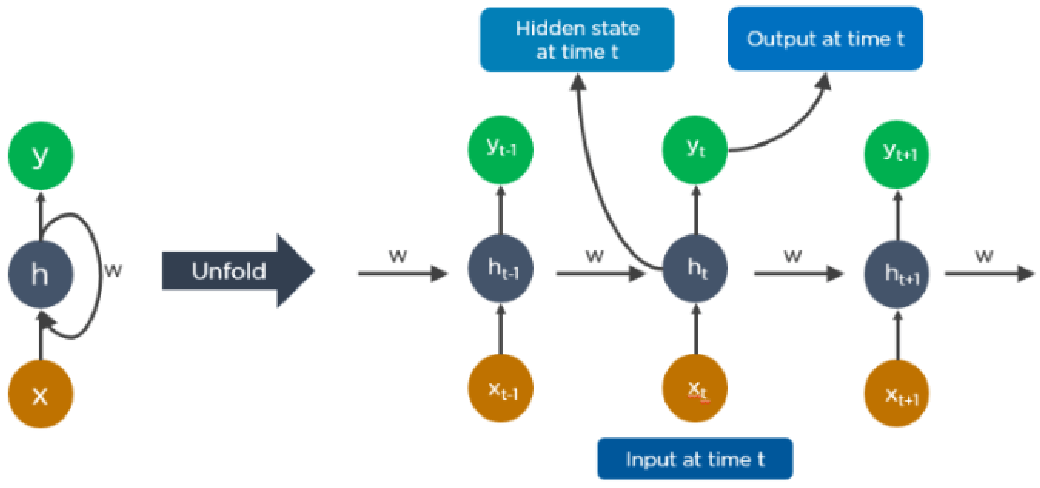


Fig15: Flowchart of RNN

4. Generative Adversarial Networks (GANs)

GANs are defined as deep learning algorithms that are used to generate new instances of data that match the training data. GAN usually consists of two components namely a generator that learns to generate false data and a discriminator that adapts itself by learning from this false data. Over some time, GANs have gained immense usage since they are frequently being used to clarify astronomical images and simulate lensing the gravitational dark matter. It is also used in video games to increase graphics for 2D textures by recreating them in higher resolution like 4K. They are also used in creating realistic cartoons character and also rendering human faces and 3D object rendering.

GANs work in simulation by generating and understanding the fake data and the real data. During the training to understand these data, the generator produces different kinds of fake data where the discriminator quickly learns to adapt and respond to it as false data. GANs then send these recognized results for updating. Consider the below image to visualize the functioning.

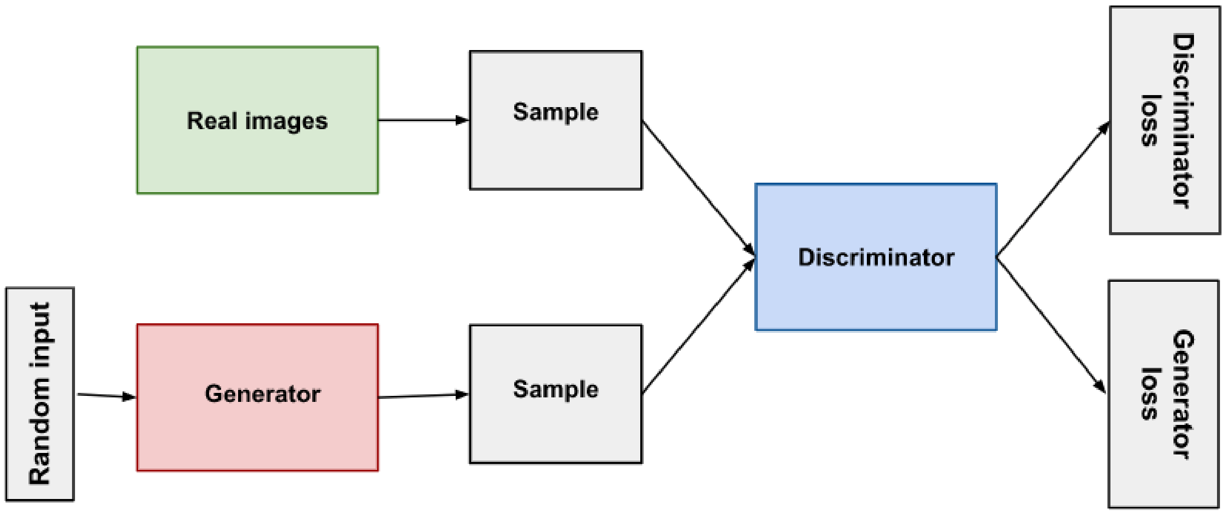


Fig16: Structure of GAN.

1. Radial Basis Function Networks (RBFNs)

RBFNs are specific types of neural networks that follow a feed-forward approach and make use of radial functions as activation functions. They consist of three layers namely the input layer, hidden layer, and output layer which are mostly used for time-series prediction, regression testing, and classification.

RBFNs do these tasks by measuring the similarities present in the training data set. They usually have an input vector that feeds these data into the input layer thereby confirming the identification and rolling out results by comparing previous data sets. Precisely, the input layer has neurons that are sensitive to these data and the nodes in the layer are efficient in classifying the class of data. Neurons are originally present in the hidden layer though they work in close integration with the input layer. The hidden layer contains Gaussian transfer functions that are inversely proportional to the distance of the output from the neuron's center. The output layer has linear combinations of the radial-based data here the Gaussian functions are passed in the neuron as parameter and output is generated. Consider the given image below to understand the process thoroughly.

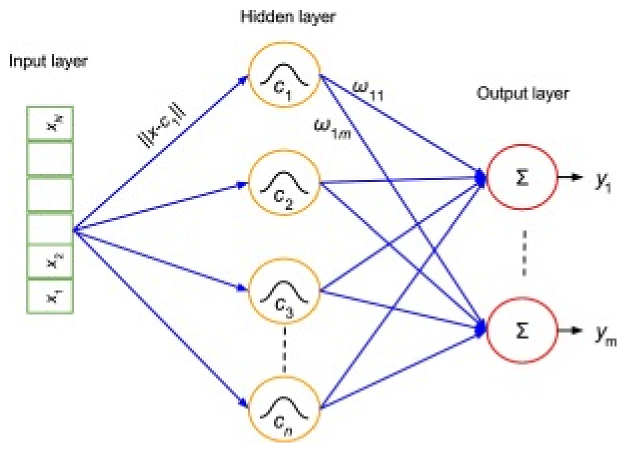


Fig17: Layers of RBFN’s

6. Multilayer Perceptrons (MLPs)

MLPs are the base of deep learning technology. It belongs to a class of feed-forward neural networks having various layers of perceptrons. These perceptrons have various activation functions in them. MLPs also have connected input and output layers and their number is the same. Also, there's a layer that remains hidden amidst these two layers. MLPs are mostly used to build image and speech recognition systems or some other types of the translation software.

The working of MLPs starts by feeding the data in the input layer. The neurons present in the layer form a graph to establish a connection that passes in one direction. The weight of this input data is found to exist between the hidden layer and the input layer.

MLPs use activation functions to determine which nodes are ready to fire. These activation functions include tanh function, sigmoid and ReLUs. MLPs are mainly used to train the models to understand what kind of co-relation the layers are serving to achieve the desired output from the given data set. See the below image to understand better.

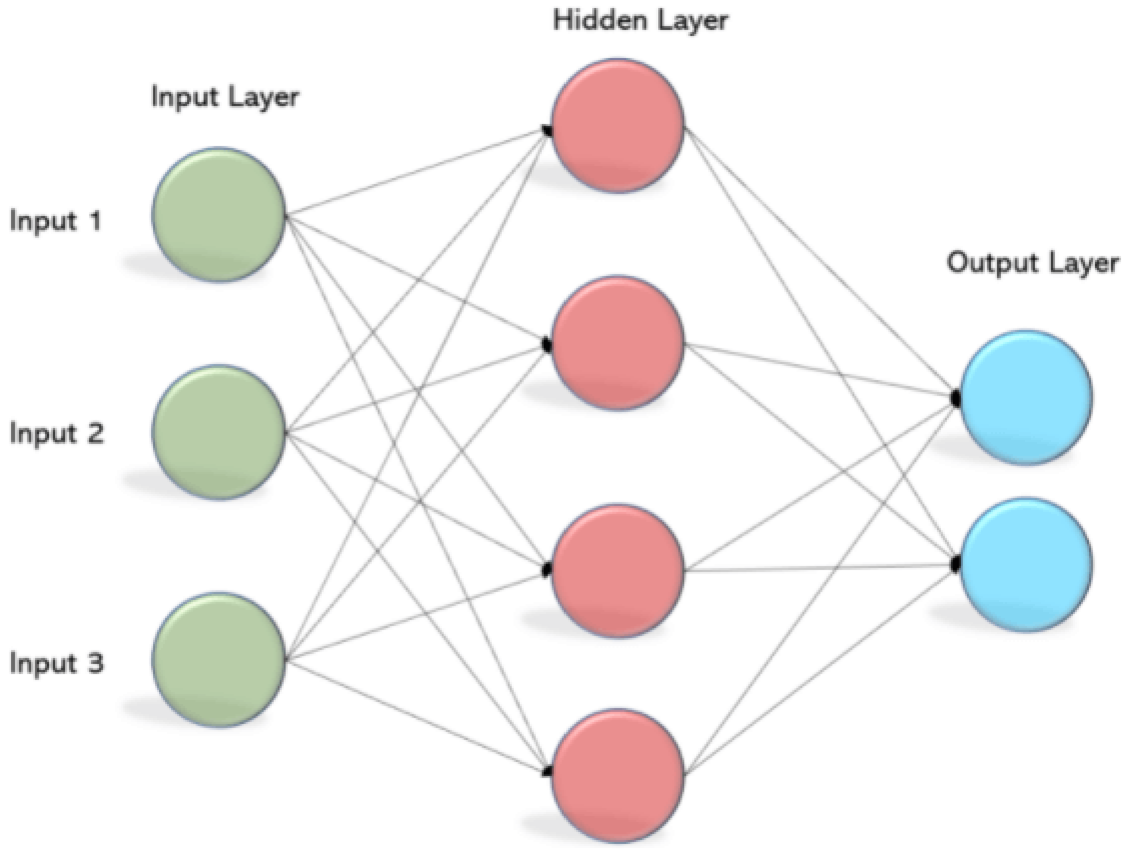


Fig18: Feed Forward Neural Network

7. Self-Organizing Maps (SOMs):

SOMs were invented by Teuvo Kohenen for achieving data visualization to understand the dimensions of data through artificial and self-organizing neural networks. The attempts to achieve data visualization to solve problems are mainly done by what humans cannot visualize. These data are generally high-dimensional so there are lesser chances of human involvement and of course less error.

SOMs help in visualizing the data by initializing weights of different nodes and then choose random vectors from the given training data. They examine each node to find the relative weights so that dependencies can be understood. The winning node is decided and that is called Best Matching Unit (BMU). Later, SOMs discover these winning nodes but the nodes reduce over time from the sample vector. So, the closer the node to BMU more is the more chance to recognize the weight and carry out further activities. There are also multiple iterations done to ensure that no node closer to BMU is missed. One example of such is the RGB color combinations that we use in our daily tasks. Consider the below image to understand how they function.

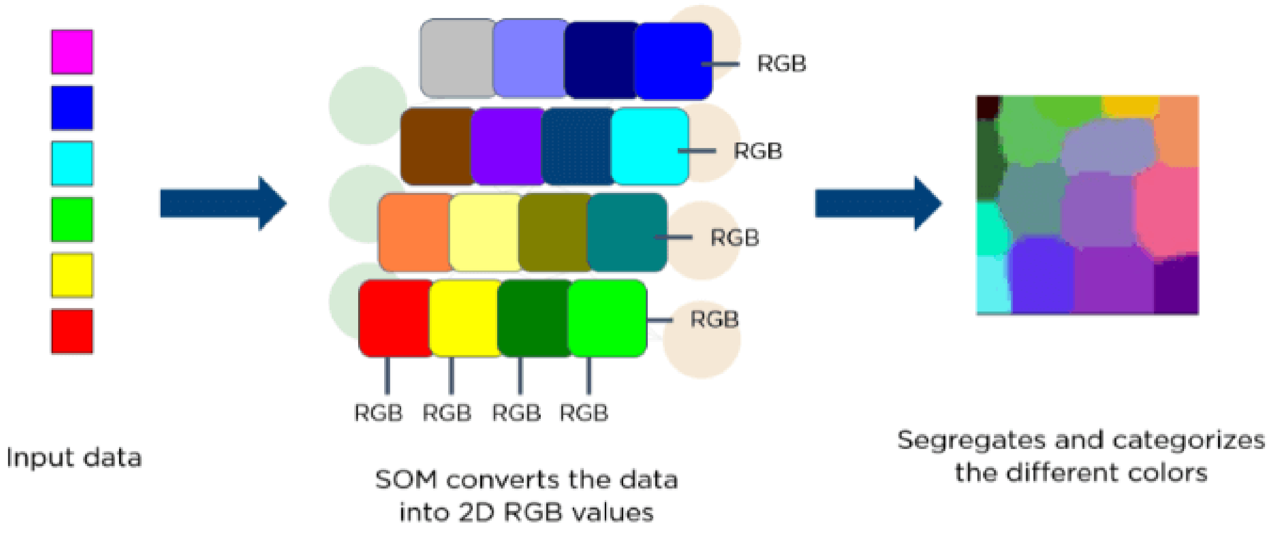

RGB
RGB
RGB
RGB
RGB RGB RGB RGB
Input data
SOM converts the data into 2D RGB values
Segregates and categorizes the different colors

Fig19: Representation of SOM's

8. Deep Belief Networks (DBNs)
DBNs are called generative models because they have various layers of latent as well as stochastic variables. The latent variable is called a hidden unit because they have binary values. DBNs are also called Boltzmann Machines because the RGM layers are stacked over each other to establish communication with previous and consecutive layers. DBNs are used in applications like video and image recognition as well as capturing motional objects.

DBNs are powered by Greedy algorithms. The layer to layer approach by leaning through a top-down approach to generate weights is the most common way DBNs function. DBNs use step by step approach of Gibbs sampling on the hidden two-layer at the top. Then, these stages draw a sample from the visible units using a model that follows the ancestral sampling method. DBNs learn from the values present in the latent value from every layer following the bottom-up pass approach.

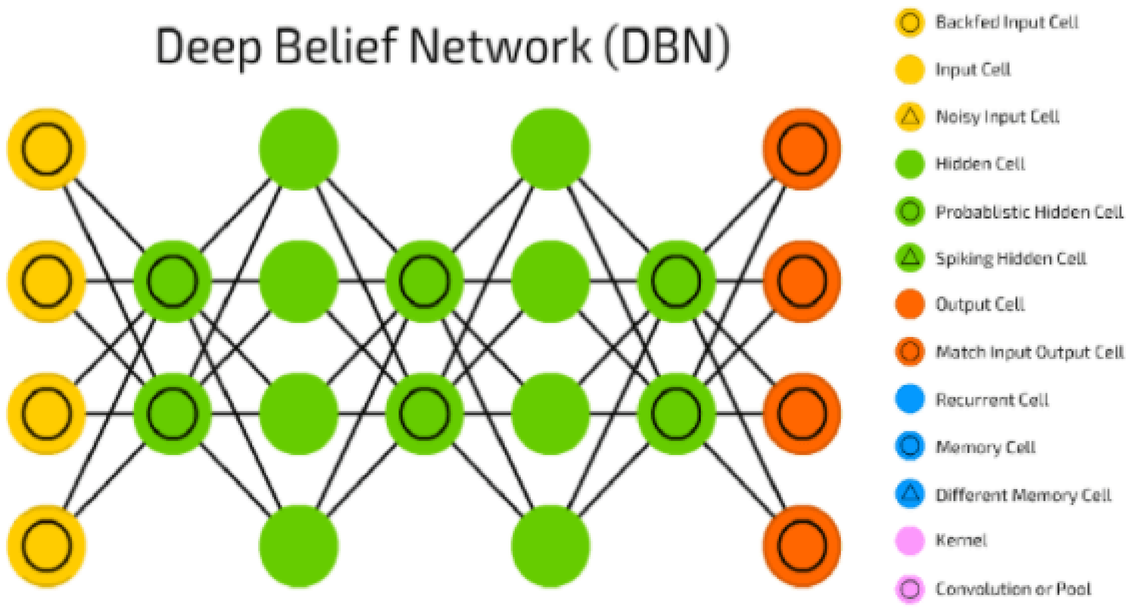


Fig20: Structure of DBN

9. Restricted Boltzmann Machines (RBMs)
RBMs were developed by Geoffrey Hinton and resemble stochastic neural networks that learn from the probability distribution in the given input set. This algorithm is mainly used in the field of dimension reduction, regression and classification, topic modeling and are considered the building blocks of DBNs. RBIs consist of two layers namely the visible layer and the hidden layer. Both of these layers are connected through hidden units and have bias units connected to nodes that generate the output. Usually, RBMs have two phases namely forward pass and backward pass.

The functioning of RBMs is carried out by accepting inputs and translating them to numbers so that inputs are encoded in the forward pass. RBMs take into account the weight of every input, and the backward pass takes these input weights and translates them further into reconstructed inputs. Later, both of these translated inputs, along with individual weights, are combined. These inputs are then pushed to the visible layer where the activation is carried out, and output is generated that can be easily reconstructed. To understand this process, consider the below image.

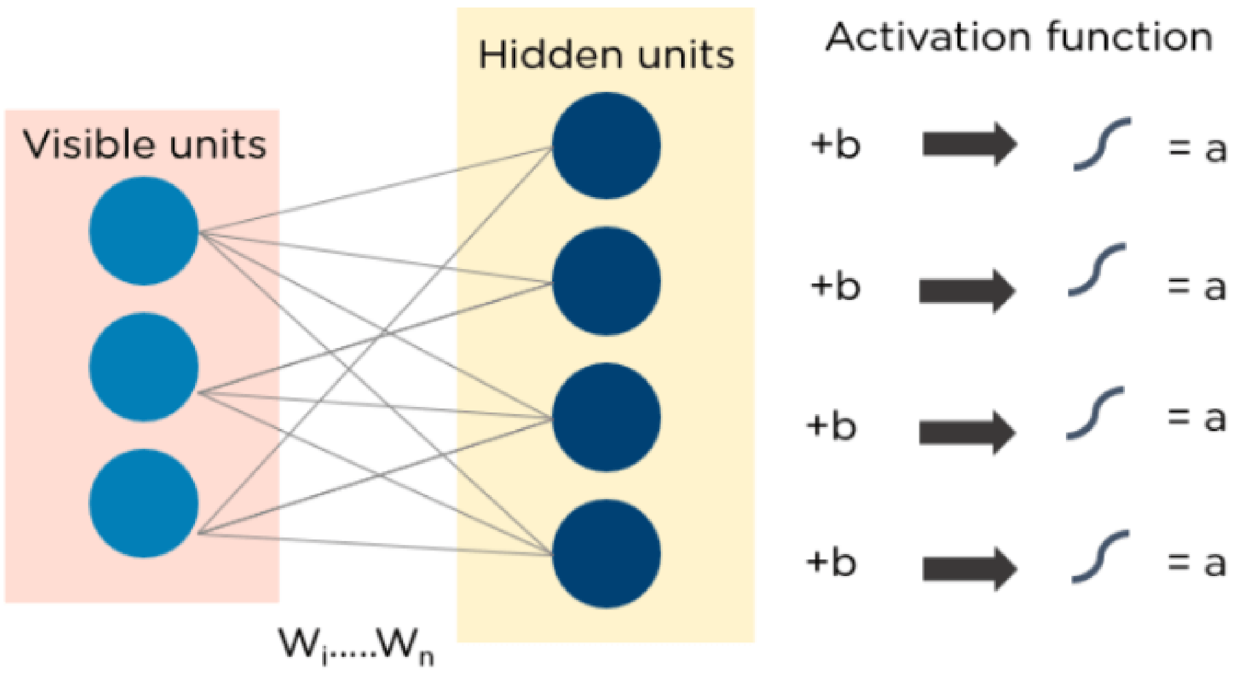


Fig21: Structure of RBN's

10. Autoencoders

Autoencoders are a special type of neural network where inputs are outputs are found usually identical. It was designed to primarily solve the problems related to unsupervised learning. Autoencoders are highly trained neural networks that replicate the data. It is the reason why the input and output are generally the same. They are used to achieve tasks like pharma discovery, image processing, and population prediction.

Autoencoders constitute three components namely the encoder, the code, and the decoder. Autoencoders are built in such a structure that they can receive inputs and transform them into various representations. The attempts to copy the original input by reconstructing them is more accurate. They do this by encoding the image or input, reduce the size. If the image is not visible properly they are passed to the neural network for clarification. Then, the clarified image is termed a reconstructed image and this resembles as accurate as of the previous image. To understand this complex process, see the below-provided image.

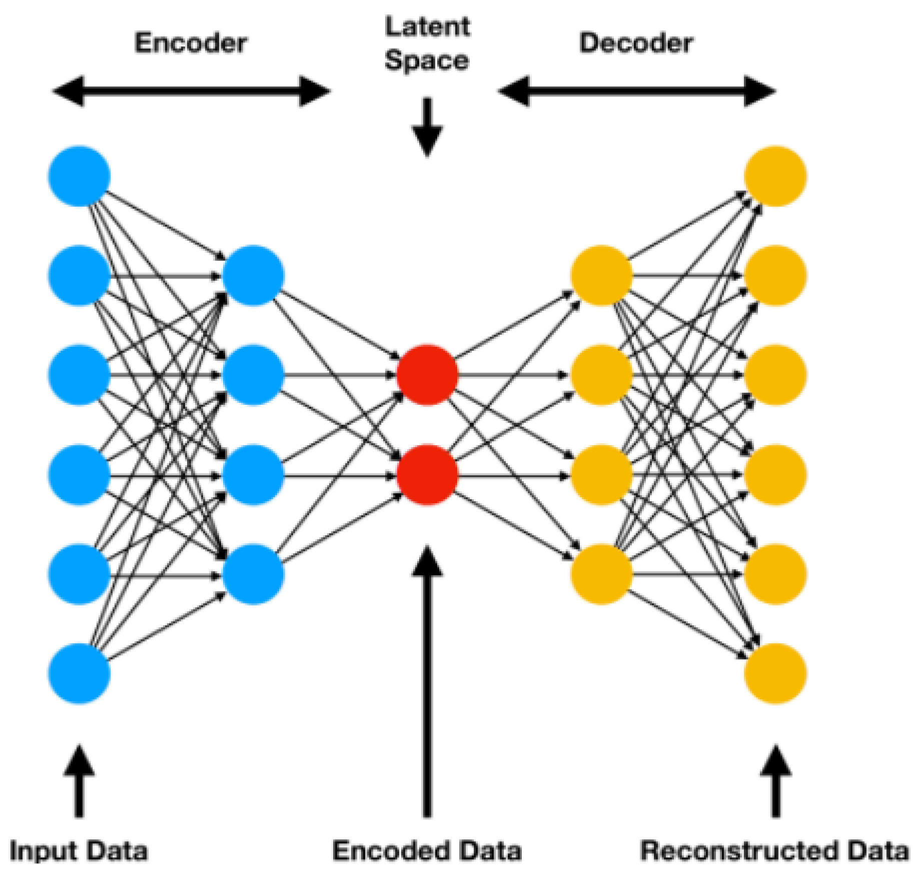


Fig22: Structure of Autoencoders

What is a CNN?

In deep learning, a convolutional neural network (CNN/ConvNet) is a class of deep neural networks, most commonly applied to analyze visual imagery. Now when we think of a neural network we think about matrix multiplications but that is not the case with ConvNet. It uses a special technique called Convolution. Now in mathematics convolution is a mathematical operation on two functions that produces a third function that expresses how the shape of one is modified by the other.
But we don’t really need to go behind the mathematics part to understand what a CNN is or how it works.

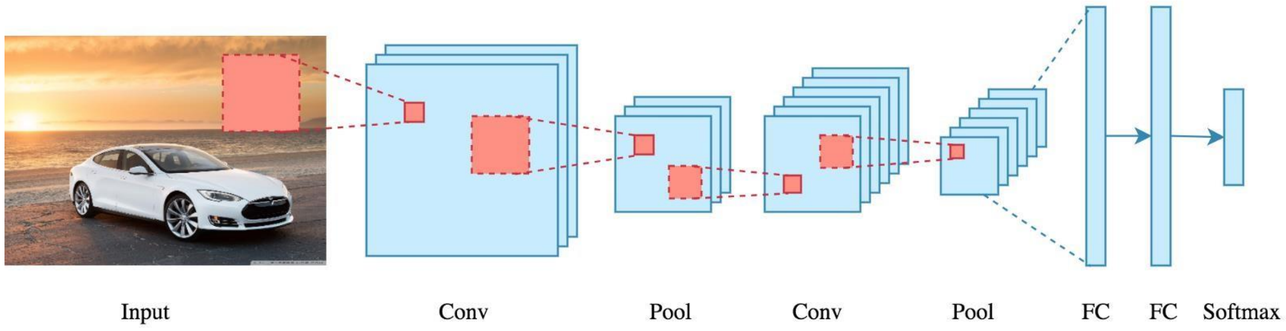


Fig23: Layers of CNN

Bottom line is that the role of the ConvNet is to reduce the images into a form that is easier to process, without losing features that are critical for getting a good prediction.

How does it work?
Before we go to the working of CNN's let's cover the basics such as what is an image and how is it represented. An RGB image is nothing but a matrix of pixel values having three planes whereas a grayscale image is the same but it has a single plane. Take a look at this image to understand more.

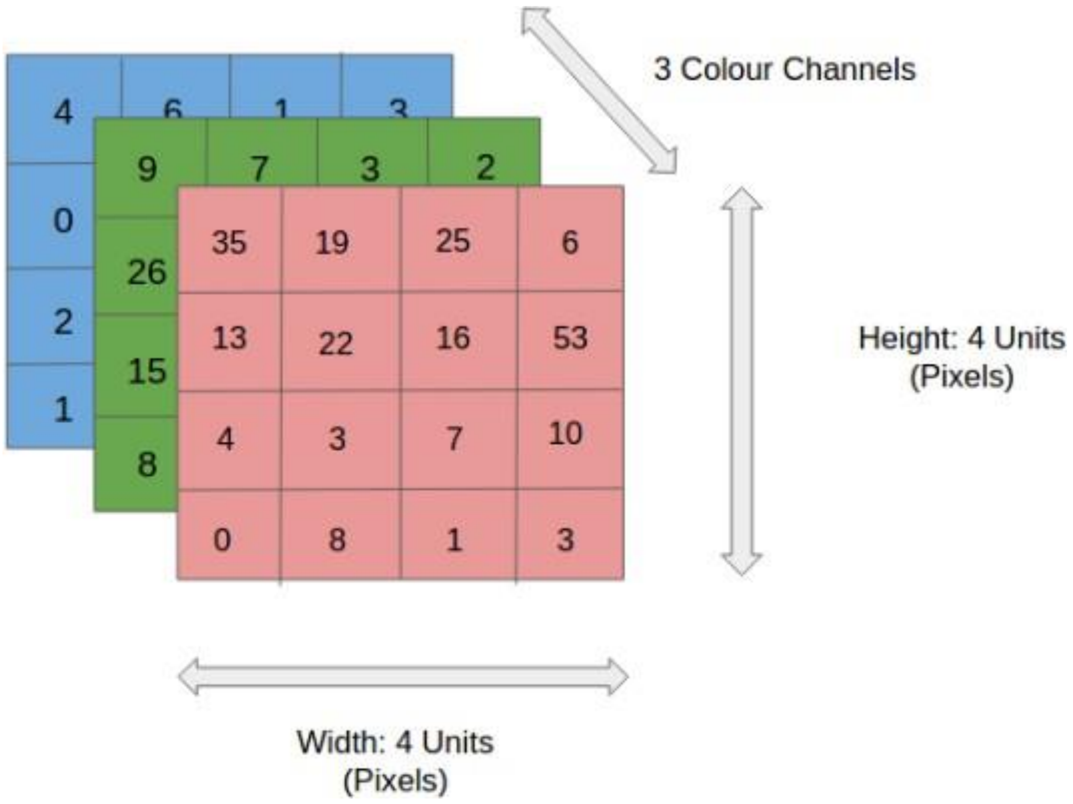


Fig24: CNN Layer

For simplicity, let's stick with grayscale images as we try to understand how CNNs work.

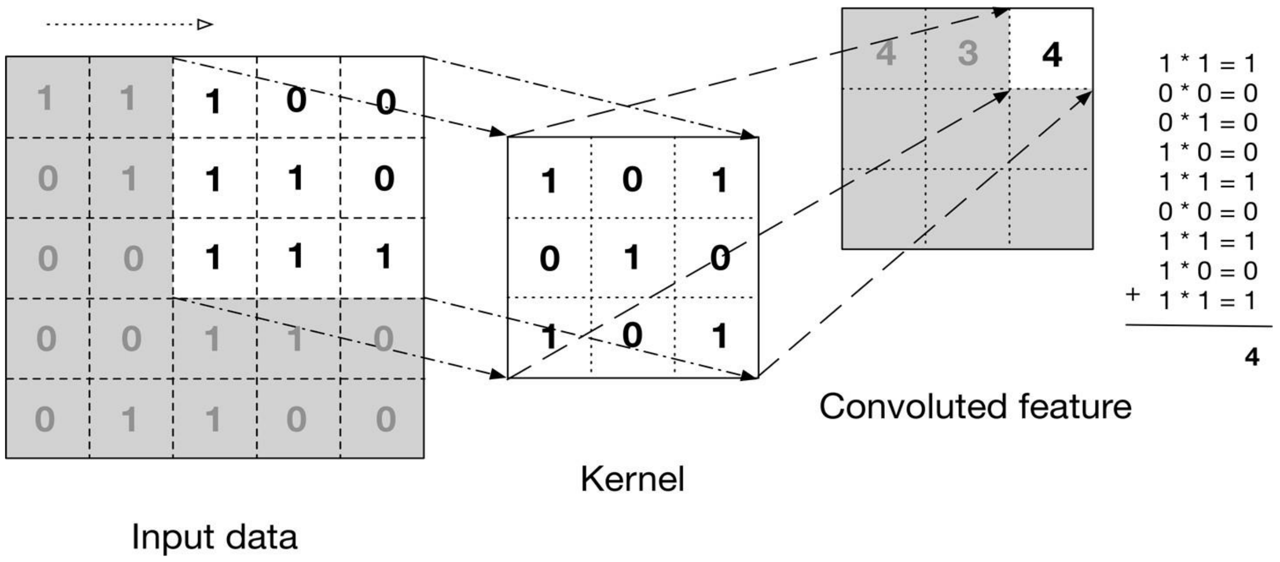


Fig25: Kernal Mapping

The above image shows what a convolution is. We take a filter/kernel (3×3 matrix) and apply it to the input image to get the convolved feature. This convolved feature is passed on to the next layer.

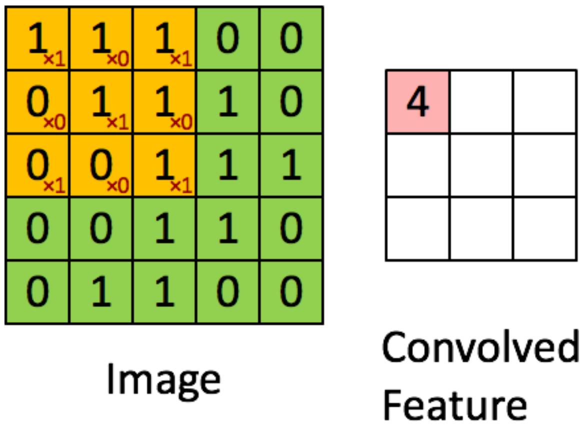


Fig26: Output for convoluted Features

In the case of RGB color, channel take a look at this animation to understand its working

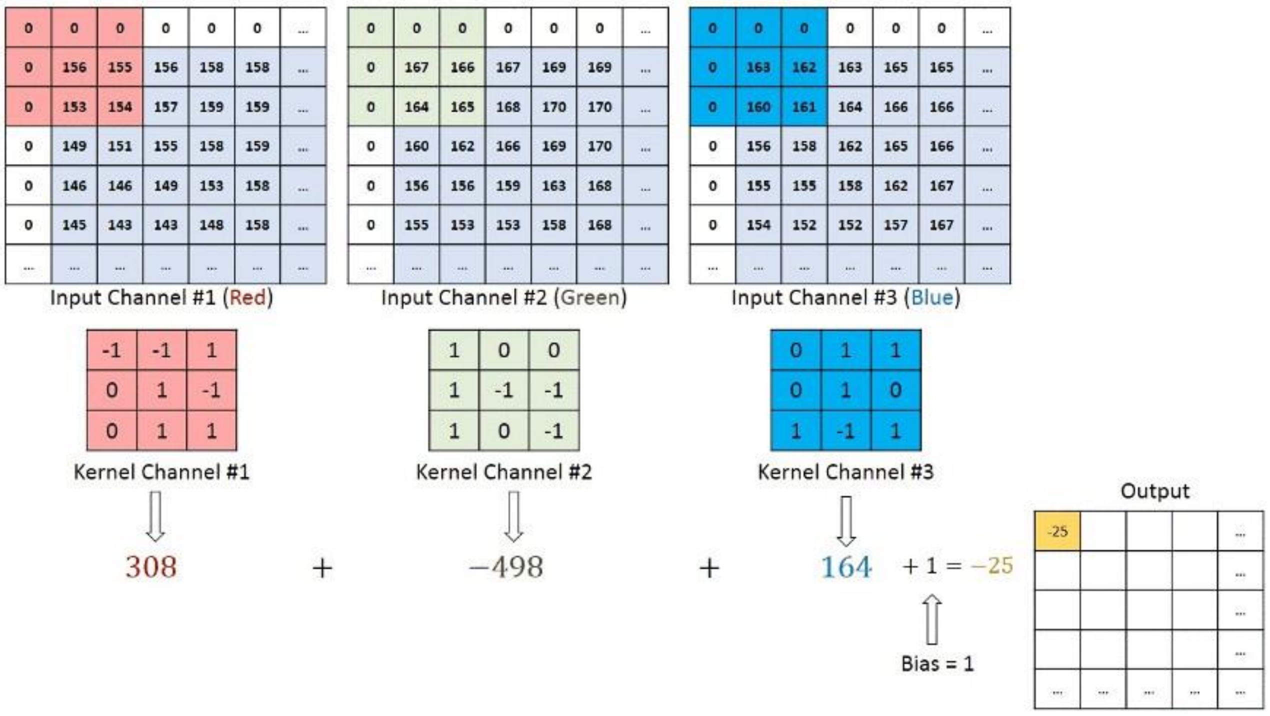


Fig27: Mapping of Kernels

Convolutional neural networks are composed of multiple layers of artificial neurons. Artificial neurons, a rough imitation of their biological counterparts, are mathematical functions that calculate the weighted sum of multiple inputs and outputs an activation value. When you input an image in a ConvNet, each layer generates several activation functions that are passed on to the next layer.

The first layer usually extracts basic features such as horizontal or diagonal edges. This output is passed on to the next layer which detects more complex features such as corners or combinational edges. As we move deeper into the network it can identify even more complex features such as objects, faces, etc.

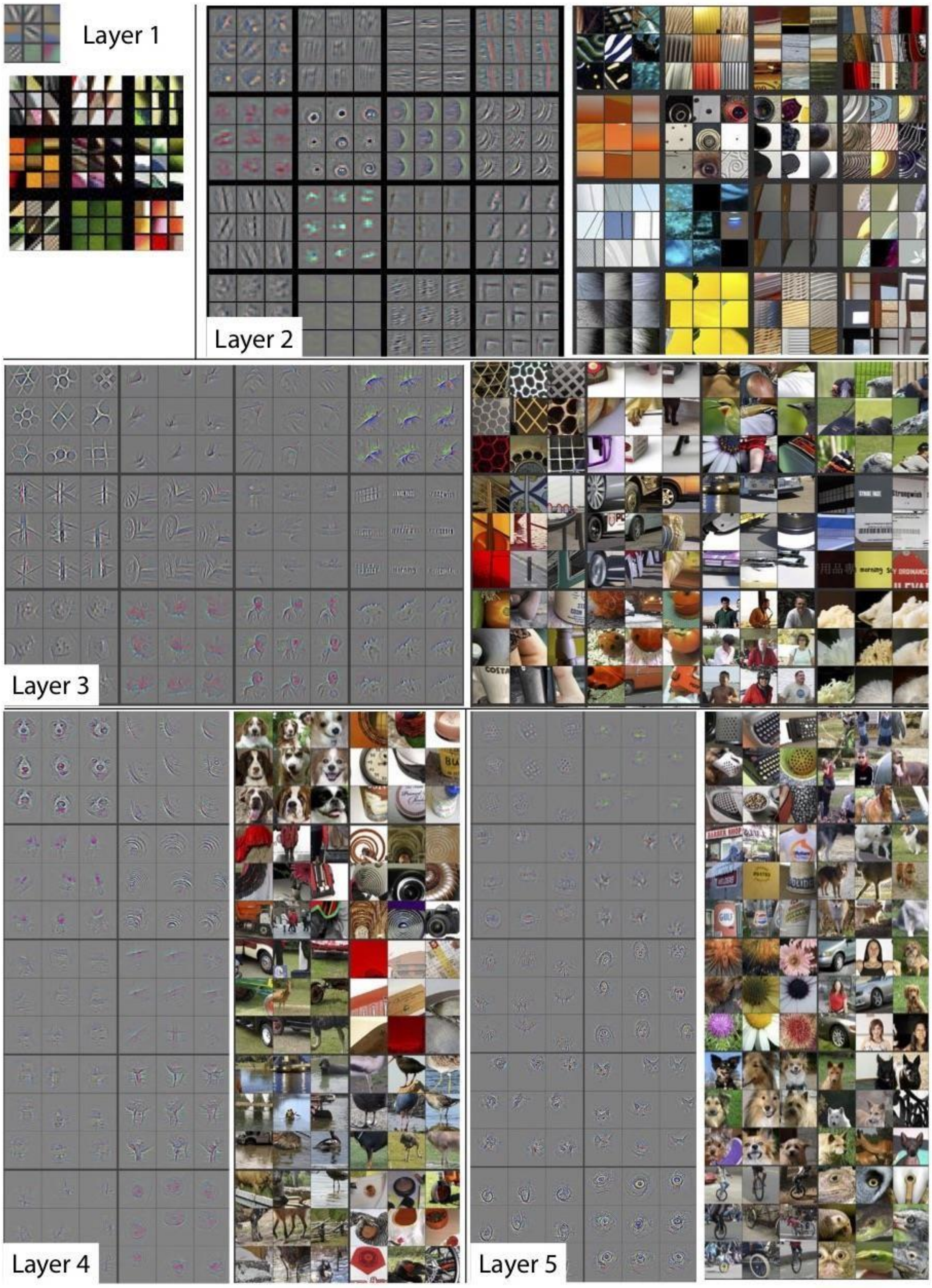


Fig28: Layers of the given model

Based on the activation map of the final convolution layer, the classification layer outputs a set of confidence scores (values between 0 and 1) that specify how likely the image is to belong to a "class." For instance, if you have a ConvNet that detects cats, dogs, and horses, the output of the final layer is the possibility that the input image contains any of those animals.

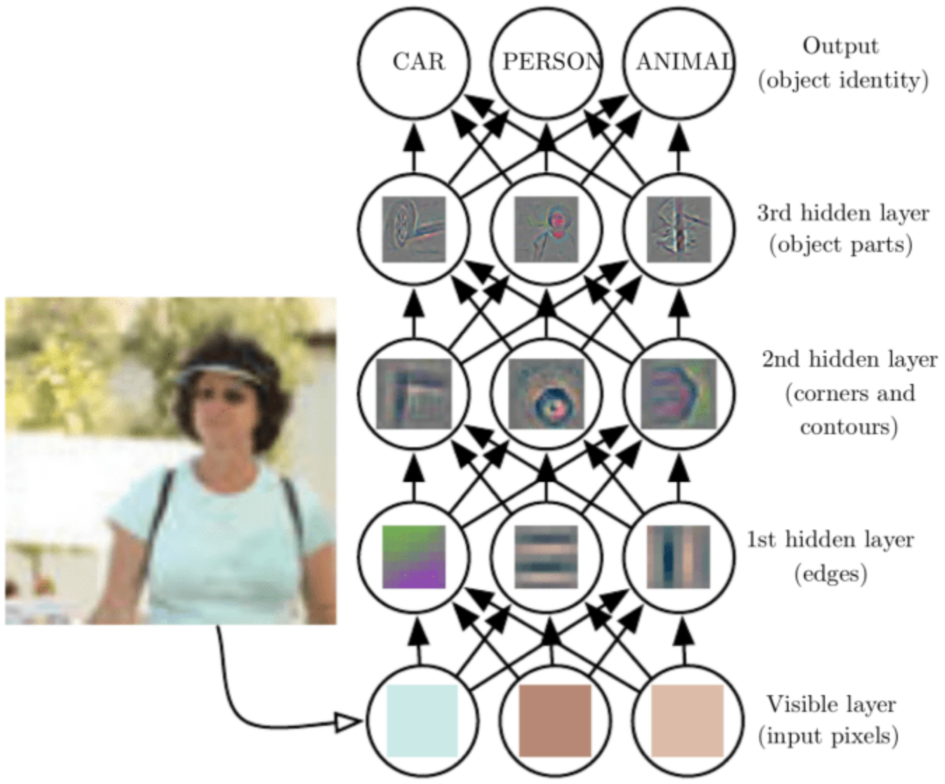

CAR
PERSON
ANIMAL
Output (object identity)
3rd hidden layer (object parts)
2nd hidden layer (corners and contours)
1st hidden layer (edges)
Visible layer (input pixels)

Fig29: Flow chart for input layer to output

What Is a Pooling Layer?

Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data by reducing the dimensions. There are two types of pooling average pooling and max pooling. I've only had experience with Max Pooling so far I haven't faced any difficulties.

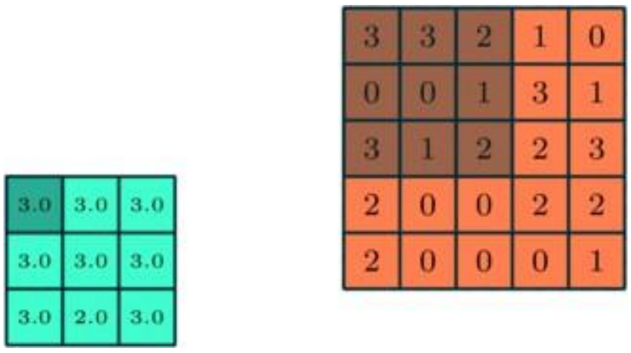


Fig30: Pooling layer

So what we do in Max Pooling is we find the maximum value of a pixel from a portion of the image covered by the kernel. Max Pooling also performs as a Noise Suppressant. It discards the noisy activations altogether and also performs de-noising along with dimensionality reduction.

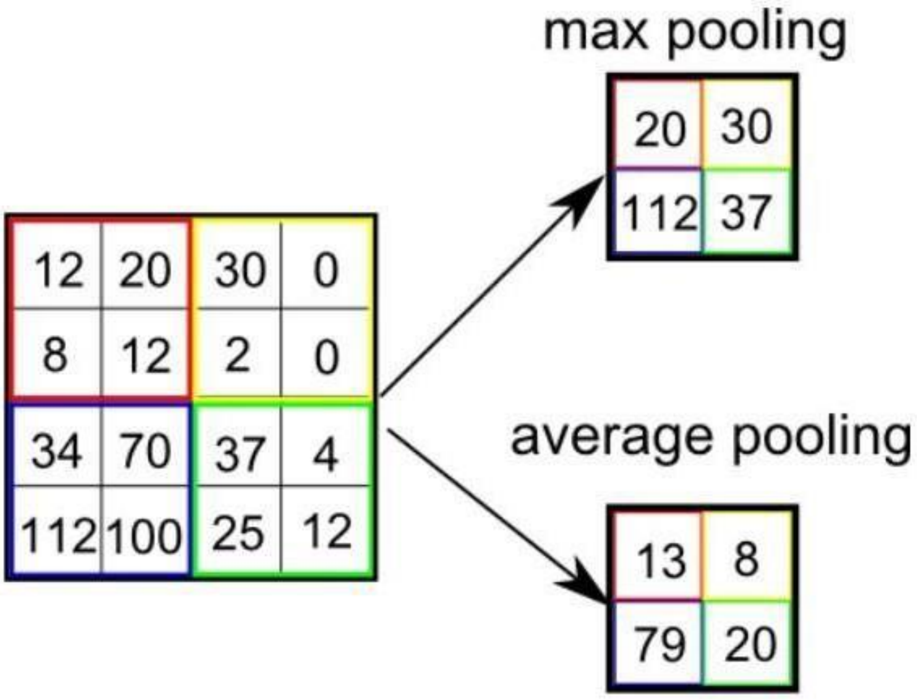


Fig31: Types of Pooling

On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel. Average Pooling simply performs dimensionality reduction as a noise suppressing mechanism. Hence, we can say that Max Pooling performs a lot better than Average Pooling.

3.3.METHODOLOGY

In this study, there are primarily 2 stages to it. Every stage is so vital and helps in building an efficient model. Following below are the steps:

A. Model Selection

B. Model Evaluation.

A. Model Selection

Here different machine learning models are used. they are as follows:

1) Logistic Regression

2) Support Vector Machine

3) Bernoulli Naïve Bayes

4) Random Forest

5) K Nearest Neighbor

6) Linear Discriminant Analysis

1). In this project, the first model used for the outcome of the project is the Logistic regression. Logistic regression is used to classify the labels from the given data. Logistic regression is basically Boolean type model where it says the input belongs to label 1 or label 2. The extracted features are loaded into this model and it is being processed and output is given. First using train_test_split function our data is split into two parts which are used for training the data and testing the data. It is split into 80:20 percent because it is the ideal split for logistic regression. Using the features logistic regression is trained by 80 percent of the data and it stores the features like specific eyes and mouth ratios f. Further in testing it uses those features which are the main ratios for that recognition.

2). Next used is SVM classifier which uses the kernel trick. Kernel trick finds the optimal line of the entire data. It classifies the data by creating a hyperplane and labels are opposite side of the hyperplane. If a data point is given for testing it decides which side it should lie.

3). Bernoulli Naive Bayes is one of the variants of the Naive Bayes algorithm in machine learning. It is very useful to be used when the dataset is in a binary distribution where the output label is either present or absent.

4). Random Forest is a powerful classifier because it uses decision trees and ensemble learning. It uses numerous classifiers to classify the data. It splits the data randomly and averages the inside models.

5). K Nearest Neighbor classifier also used to analyze how it is behaving for the given data since it is also widely used classifier. It uses proximity for classification.

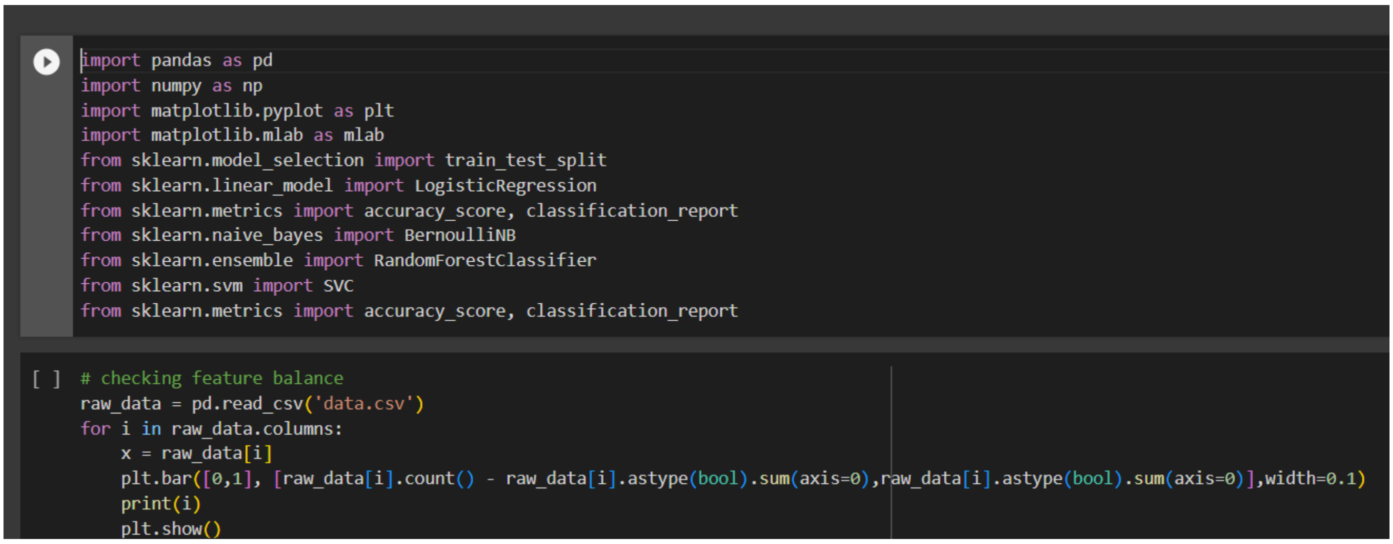
6). Linear Discriminant Analysis is a recent classifier. It generally used for dimensionality reduction and as a classifier. It is used for versatility of the project.

B-Model Evaluation

Model evaluation is the final stage of the project. This shows the results by calculating some of the main parameters for a model. It calculates the accuracy, confusion matrix, precision score, recall, F1 score and so on. The key parameters are accuracy and confusion matrix. Both majorly signifies how the model is working first confusion matrix is calculated using formulas. Using the information from confusion matrix accuracy is measured. Accuraczy says how well the model is classifying the input tweets or the text.

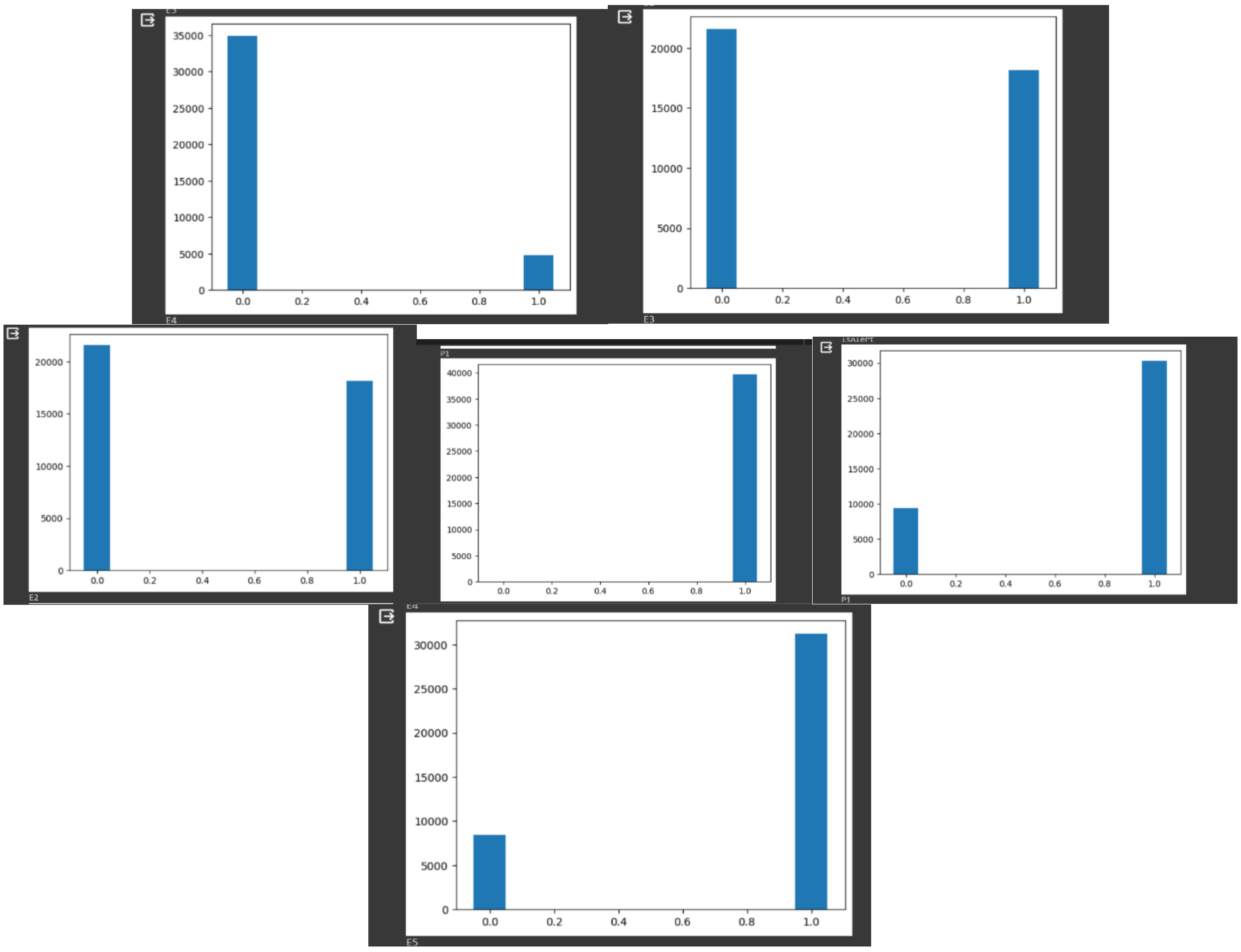
Chapter – 4

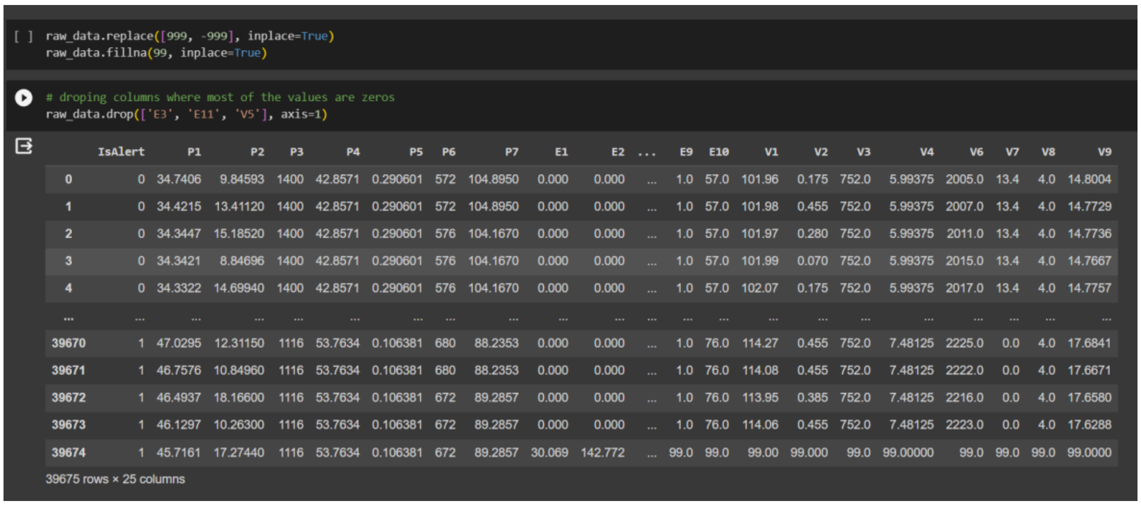
IMPLEMENTATION SNAPSHOTS OF SOURCE CODE



```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.mlab as mlab
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
from sklearn.naive_bayes import BernoulliNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report
```

```
[ ]  # checking feature balance
     raw_data = pd.read_csv('data.csv')
     for i in raw_data.columns:
         x = raw_data[i]
         plt.bar([0,1], [raw_data[i].count() - raw_data[i].astype(bool).sum(axis=0),raw_data[i].astype(bool).sum(axis=0)],width=0.1)
         print(i)
         plt.show()
```



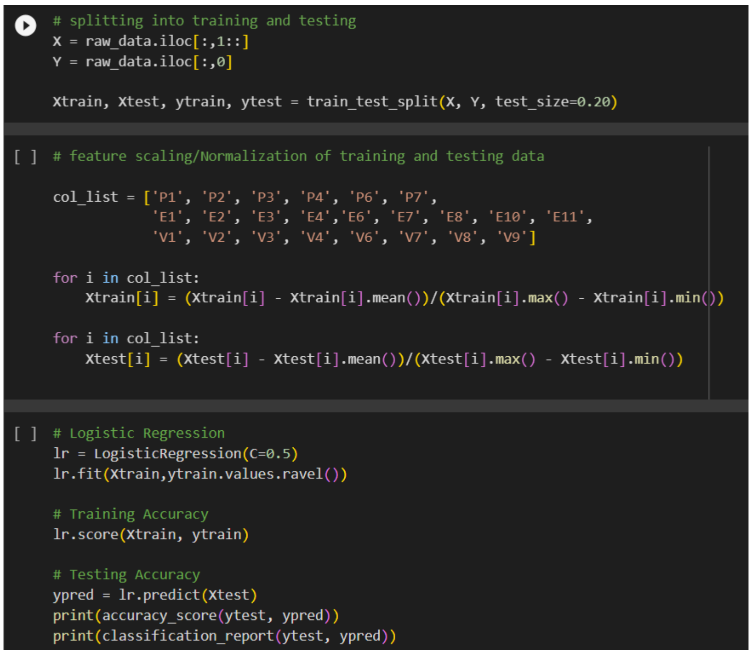


```
[ ] raw_data.replace([999, -999], inplace=True)
    raw_data.fillna(99, inplace=True)
```

```
# droping columns where most of the values are zeros
raw_data.drop(['E3', 'E11', 'V5'], axis=1)
```

	IsAlert	P1	P2	P3	P4	P5	P6	P7	E1	E2	...	E9	E10	V1	V2	V3	V4	V6	V7	V8	V9
0	0	34.7406	9.84593	1400	42.8571	0.290601	572	104.8950	0.000	0.000	...	1.0	57.0	101.96	0.175	752.0	5.99375	2005.0	13.4	4.0	14.8004
1	0	34.4215	13.41120	1400	42.8571	0.290601	572	104.8950	0.000	0.000	...	1.0	57.0	101.98	0.455	752.0	5.99375	2007.0	13.4	4.0	14.7729
2	0	34.3447	15.18520	1400	42.8571	0.290601	576	104.1670	0.000	0.000	...	1.0	57.0	101.97	0.280	752.0	5.99375	2011.0	13.4	4.0	14.7736
3	0	34.3421	8.84696	1400	42.8571	0.290601	576	104.1670	0.000	0.000	...	1.0	57.0	101.99	0.070	752.0	5.99375	2015.0	13.4	4.0	14.7667
4	0	34.3322	14.69940	1400	42.8571	0.290601	576	104.1670	0.000	0.000	...	1.0	57.0	102.07	0.175	752.0	5.99375	2017.0	13.4	4.0	14.7757
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
39670	1	47.0295	12.31150	1116	53.7634	0.106381	680	88.2353	0.000	0.000	...	1.0	76.0	114.27	0.455	752.0	7.48125	2225.0	0.0	4.0	17.6841
39671	1	46.7576	10.84960	1116	53.7634	0.106381	680	88.2353	0.000	0.000	...	1.0	76.0	114.08	0.455	752.0	7.48125	2222.0	0.0	4.0	17.6671
39672	1	46.4937	18.16600	1116	53.7634	0.106381	672	89.2857	0.000	0.000	...	1.0	76.0	113.95	0.385	752.0	7.48125	2216.0	0.0	4.0	17.6580
39673	1	46.1297	10.26300	1116	53.7634	0.106381	672	89.2857	0.000	0.000	...	1.0	76.0	114.06	0.455	752.0	7.48125	2223.0	0.0	4.0	17.6288
39674	1	45.7161	17.27440	1116	53.7634	0.106381	672	89.2857	30.069	142.772	...	99.0	99.0	99.00	99.000	99.0	99.00000	99.0	99.0	99.0	99.0000

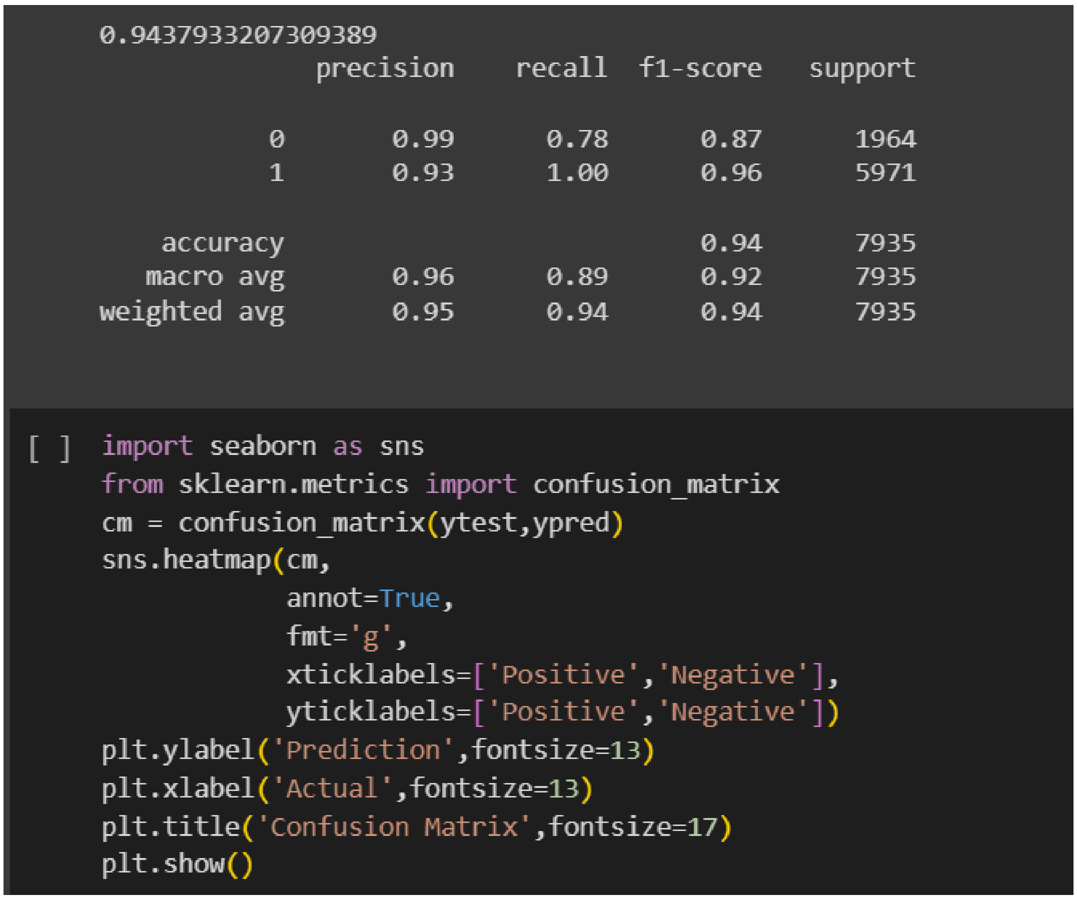
39675 rows × 25 columns



```
# splitting into training and testing
X = raw_data.iloc[:,1::]
Y = raw_data.iloc[:,0]

Xtrain, Xtest, ytrain, ytest = train_test_split(X, Y, test_size=0.20)
```

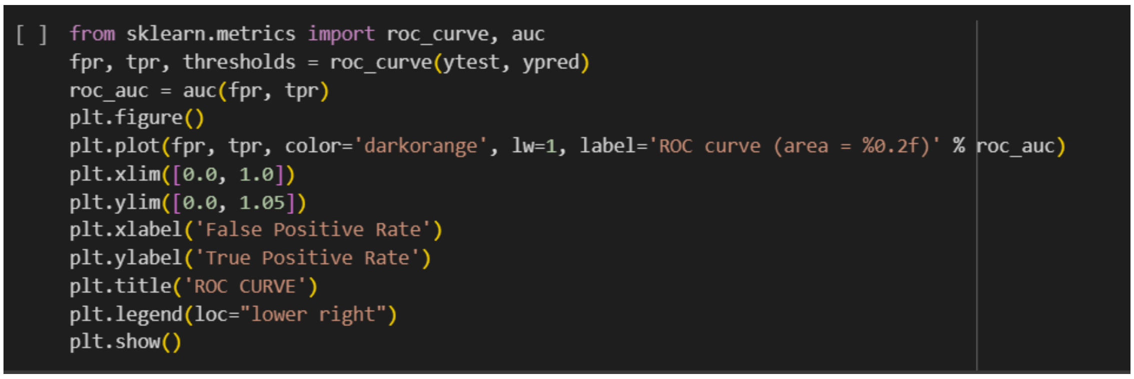
```
[ ] # feature scaling/Normalization of training and testing data

    col_list = ['P1', 'P2', 'P3', 'P4', 'P6', 'P7',
                'E1', 'E2', 'E3', 'E4','E6', 'E7', 'E8', 'E10', 'E11',
                'V1', 'V2', 'V3', 'V4', 'V6', 'V7', 'V8', 'V9']

    for i in col_list:
        Xtrain[i] = (Xtrain[i] - Xtrain[i].mean())/(Xtrain[i].max() - Xtrain[i].min())

    for i in col_list:
        Xtest[i] = (Xtest[i] - Xtest[i].mean())/(Xtest[i].max() - Xtest[i].min())
```

```
[ ] # Logistic Regression
    lr = LogisticRegression(C=0.5)
    lr.fit(Xtrain,ytrain.values.ravel())

    # Training Accuracy
    lr.score(Xtrain, ytrain)

    # Testing Accuracy
    ypred = lr.predict(Xtest)
    print(accuracy_score(ytest, ypred))
    print(classification_report(ytest, ypred))
```

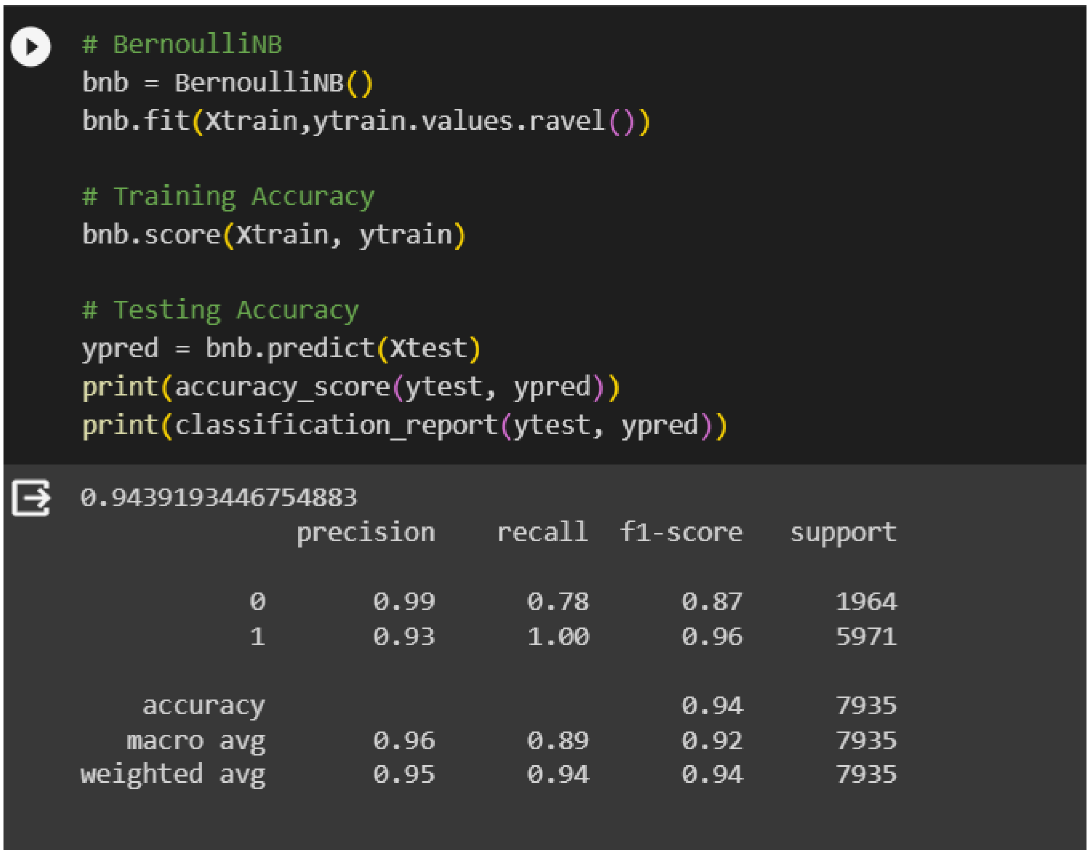


```
0.9437933207309389
              precision    recall  f1-score   support

           0       0.99      0.78      0.87      1964
           1       0.93      1.00      0.96      5971

    accuracy                           0.94      7935
   macro avg       0.96      0.89      0.92      7935
weighted avg       0.95      0.94      0.94      7935
```

```
[ ] import seaborn as sns
    from sklearn.metrics import confusion_matrix
    cm = confusion_matrix(ytest,ypred)
    sns.heatmap(cm,
                annot=True,
                fmt='g',
                xticklabels=['Positive','Negative'],
                yticklabels=['Positive','Negative'])
    plt.ylabel('Prediction',fontsize=13)
    plt.xlabel('Actual',fontsize=13)
    plt.title('Confusion Matrix',fontsize=17)
    plt.show()
```

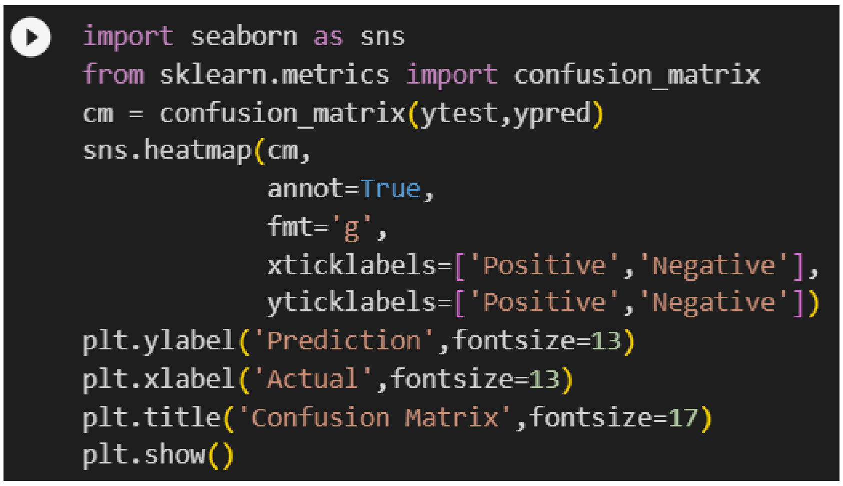


```
[ ] from sklearn.metrics import roc_curve, auc
    fpr, tpr, thresholds = roc_curve(ytest, ypred)
    roc_auc = auc(fpr, tpr)
    plt.figure()
    plt.plot(fpr, tpr, color='darkorange', lw=1, label='ROC curve (area = %0.2f)' % roc_auc)
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('ROC CURVE')
    plt.legend(loc="lower right")
    plt.show()
```

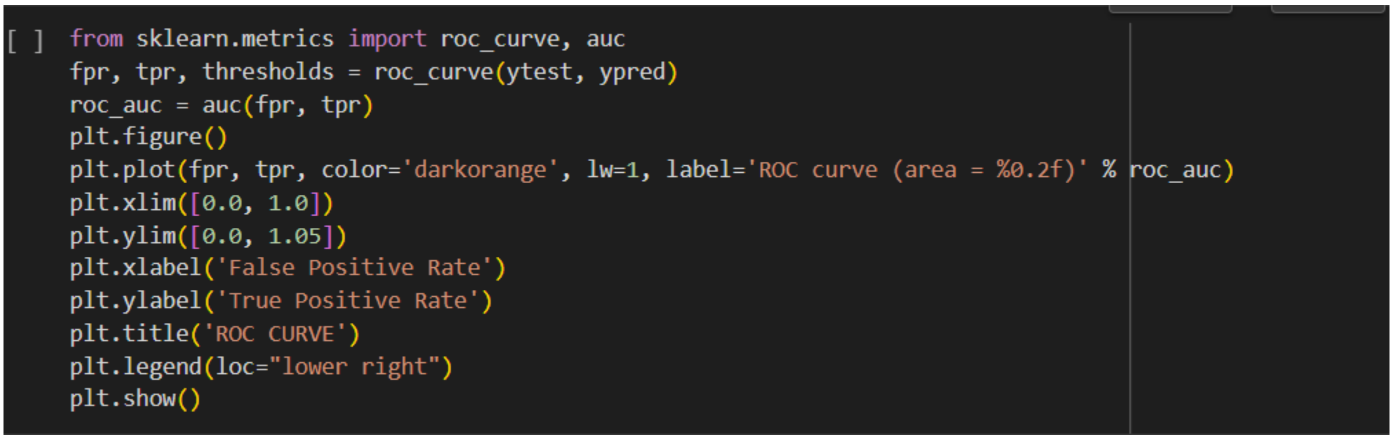


```
# BernoulliNB
bnb = BernoulliNB()
bnb.fit(Xtrain,ytrain.values.ravel())

# Training Accuracy
bnb.score(Xtrain, ytrain)

# Testing Accuracy
ypred = bnb.predict(Xtest)
print(accuracy_score(ytest, ypred))
print(classification_report(ytest, ypred))
```

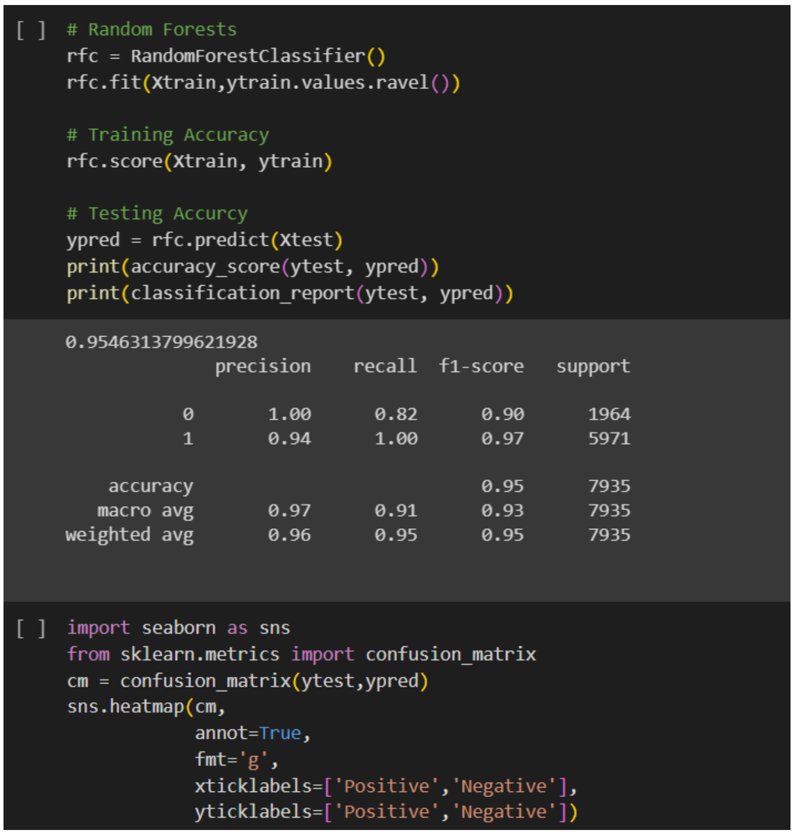
```
0.9439193446754883
              precision    recall  f1-score   support

           0       0.99      0.78      0.87      1964
           1       0.93      1.00      0.96      5971

    accuracy                           0.94      7935
   macro avg       0.96      0.89      0.92      7935
weighted avg       0.95      0.94      0.94      7935
```



```
import seaborn as sns
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(ytest,ypred)
sns.heatmap(cm,
            annot=True,
            fmt='g',
            xticklabels=['Positive','Negative'],
            yticklabels=['Positive','Negative'])
plt.ylabel('Prediction',fontsize=13)
plt.xlabel('Actual',fontsize=13)
plt.title('Confusion Matrix',fontsize=17)
plt.show()
```



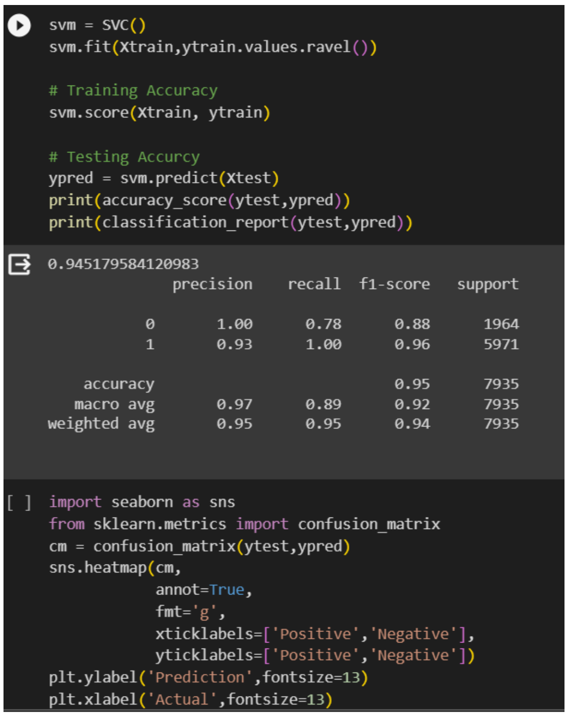
```
[ ]  from sklearn.metrics import roc_curve, auc
     fpr, tpr, thresholds = roc_curve(ytest, ypred)
     roc_auc = auc(fpr, tpr)
     plt.figure()
     plt.plot(fpr, tpr, color='darkorange', lw=1, label='ROC curve (area = %0.2f)' % roc_auc)
     plt.xlim([0.0, 1.0])
     plt.ylim([0.0, 1.05])
     plt.xlabel('False Positive Rate')
     plt.ylabel('True Positive Rate')
     plt.title('ROC CURVE')
     plt.legend(loc="lower right")
     plt.show()
```



```
[ ]  # Random Forests
     rfc = RandomForestClassifier()
     rfc.fit(Xtrain,ytrain.values.ravel())

     # Training Accuracy
     rfc.score(Xtrain, ytrain)

     # Testing Accurcy
     ypred = rfc.predict(Xtest)
     print(accuracy_score(ytest, ypred))
     print(classification_report(ytest, ypred))
```

```
     0.9546313799621928
                   precision    recall  f1-score   support

                0       1.00      0.82      0.90      1964
                1       0.94      1.00      0.97      5971

         accuracy                           0.95      7935
        macro avg       0.97      0.91      0.93      7935
     weighted avg       0.96      0.95      0.95      7935
```

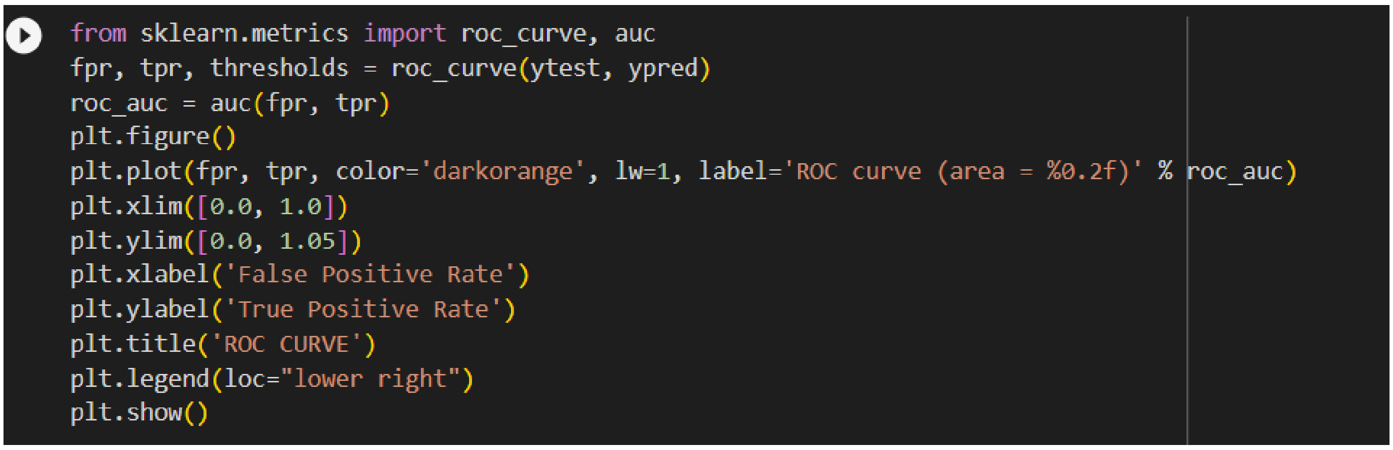
```
[ ]  import seaborn as sns
     from sklearn.metrics import confusion_matrix
     cm = confusion_matrix(ytest,ypred)
     sns.heatmap(cm,
                 annot=True,
                 fmt='g',
                 xticklabels=['Positive','Negative'],
                 yticklabels=['Positive','Negative'])
```



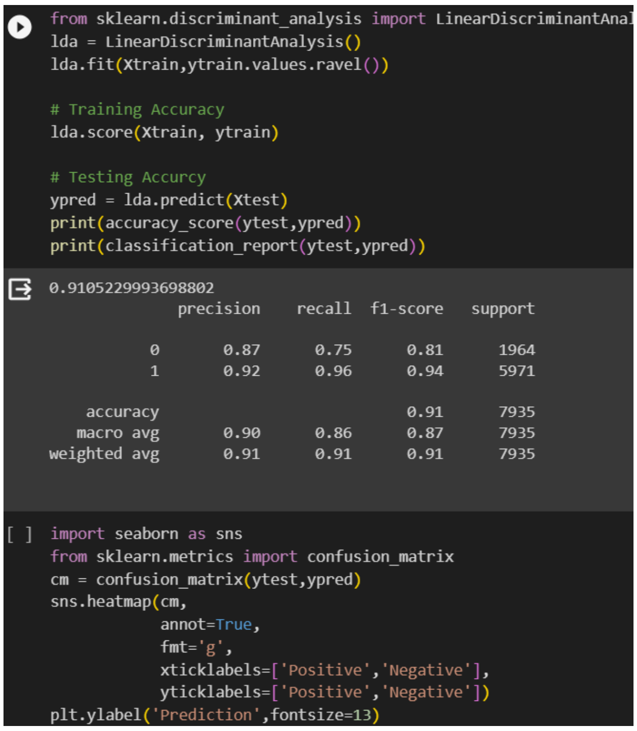
```
svm = SVC()
svm.fit(Xtrain,ytrain.values.ravel())

# Training Accuracy
svm.score(Xtrain, ytrain)

# Testing Accurcy
ypred = svm.predict(Xtest)
print(accuracy_score(ytest,ypred))
print(classification_report(ytest,ypred))
```

```
0.945179584120983
              precision    recall  f1-score   support

           0       1.00      0.78      0.88      1964
           1       0.93      1.00      0.96      5971

    accuracy                           0.95      7935
   macro avg       0.97      0.89      0.92      7935
weighted avg       0.95      0.95      0.94      7935
```

```
[ ] import seaborn as sns
    from sklearn.metrics import confusion_matrix
    cm = confusion_matrix(ytest,ypred)
    sns.heatmap(cm,
                annot=True,
                fmt='g',
                xticklabels=['Positive','Negative'],
                yticklabels=['Positive','Negative'])
    plt.ylabel('Prediction',fontsize=13)
    plt.xlabel('Actual',fontsize=13)
```



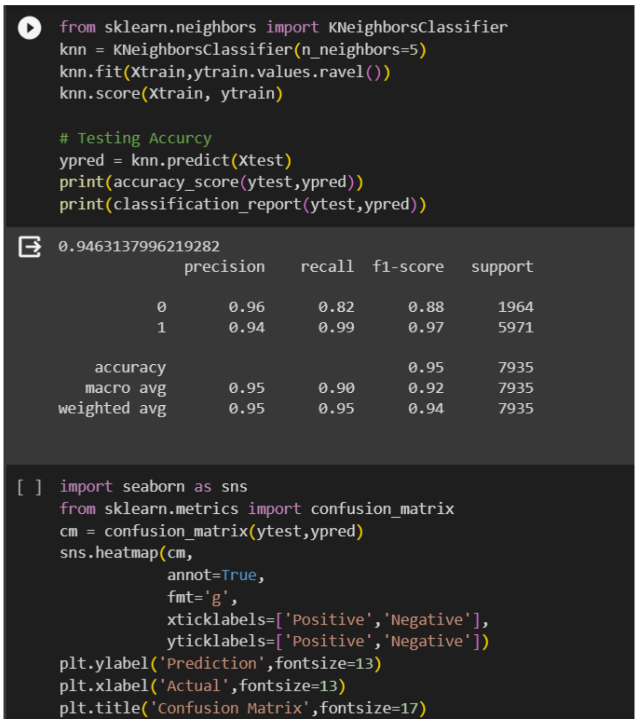
```
from sklearn.metrics import roc_curve, auc
fpr, tpr, thresholds = roc_curve(ytest, ypred)
roc_auc = auc(fpr, tpr)
plt.figure()
plt.plot(fpr, tpr, color='darkorange', lw=1, label='ROC curve (area = %0.2f)' % roc_auc)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC CURVE')
plt.legend(loc="lower right")
plt.show()
```



```
from sklearn.discriminant_analysis import LinearDiscriminantAnal
lda = LinearDiscriminantAnalysis()
lda.fit(Xtrain,ytrain.values.ravel())

# Training Accuracy
lda.score(Xtrain, ytrain)

# Testing Accurcy
ypred = lda.predict(Xtest)
print(accuracy_score(ytest,ypred))
print(classification_report(ytest,ypred))
```

```
0.9105229993698802
              precision    recall  f1-score   support

           0       0.87      0.75      0.81      1964
           1       0.92      0.96      0.94      5971

    accuracy                           0.91      7935
   macro avg       0.90      0.86      0.87      7935
weighted avg       0.91      0.91      0.91      7935
```

```
[ ] import seaborn as sns
    from sklearn.metrics import confusion_matrix
    cm = confusion_matrix(ytest,ypred)
    sns.heatmap(cm,
                annot=True,
                fmt='g',
                xticklabels=['Positive','Negative'],
                yticklabels=['Positive','Negative'])
    plt.ylabel('Prediction',fontsize=13)
```



```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(Xtrain,ytrain.values.ravel())
knn.score(Xtrain, ytrain)

# Testing Accurcy
ypred = knn.predict(Xtest)
print(accuracy_score(ytest,ypred))
print(classification_report(ytest,ypred))
```

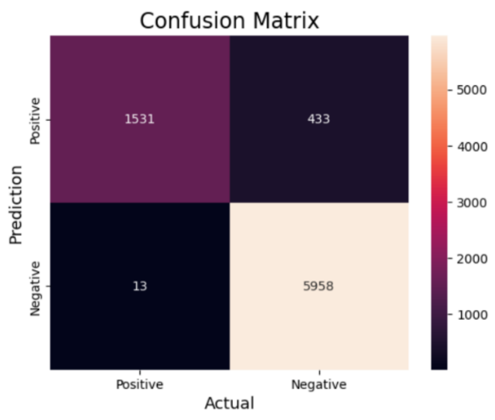
```
0.9463137996219282
              precision    recall  f1-score   support

           0       0.96      0.82      0.88      1964
           1       0.94      0.99      0.97      5971

    accuracy                           0.95      7935
   macro avg       0.95      0.90      0.92      7935
weighted avg       0.95      0.95      0.94      7935
```

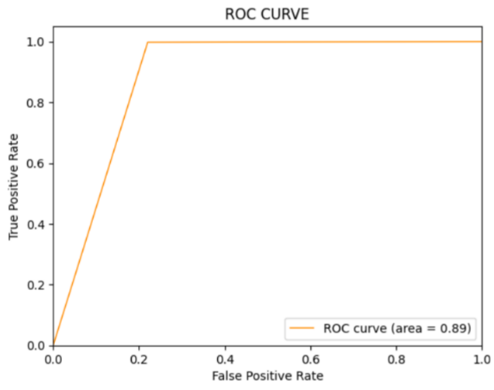
```
[ ] import seaborn as sns
    from sklearn.metrics import confusion_matrix
    cm = confusion_matrix(ytest,ypred)
    sns.heatmap(cm,
                annot=True,
                fmt='g',
                xticklabels=['Positive','Negative'],
                yticklabels=['Positive','Negative'])
    plt.ylabel('Prediction',fontsize=13)
    plt.xlabel('Actual',fontsize=13)
    plt.title('Confusion Matrix',fontsize=17)
```

Chapter – 5

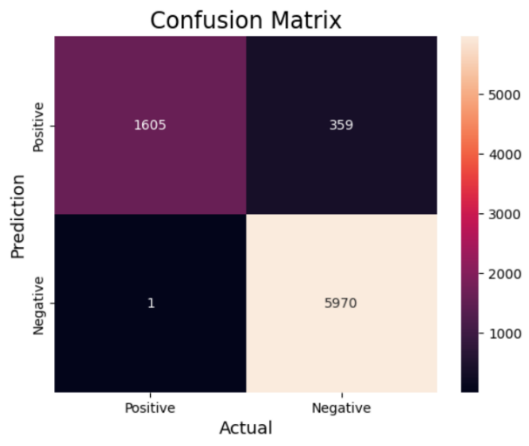
RESULT ANALYSIS AND VALIDATION

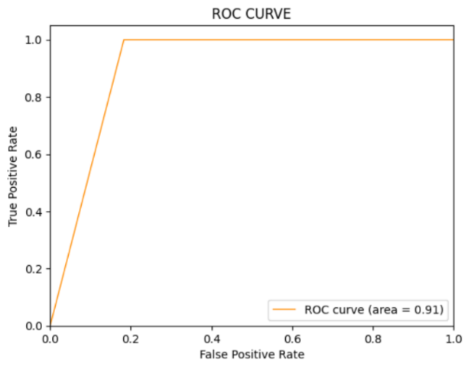
LOGISTIC REGRESSION



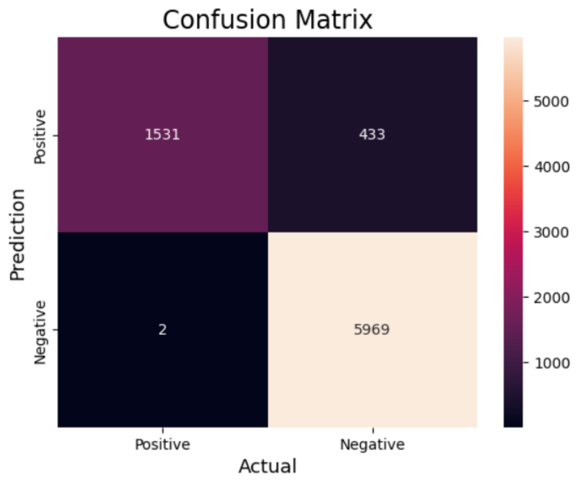


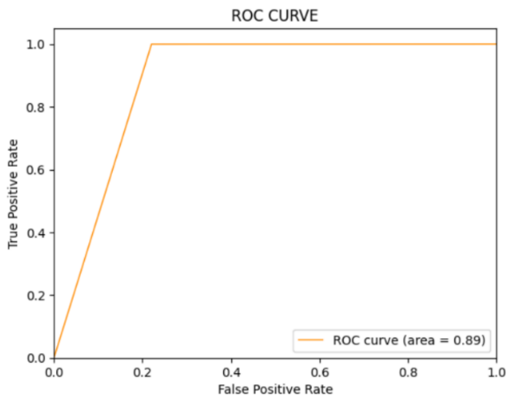
RANDOM FOREST



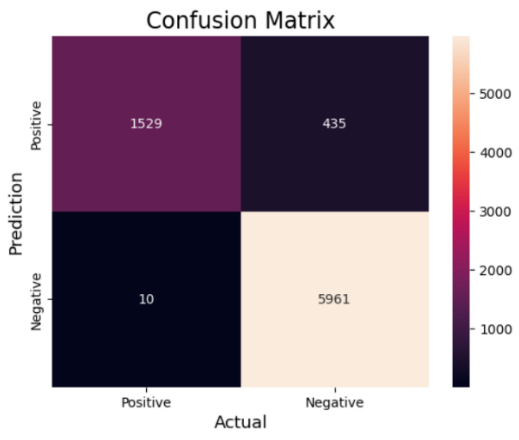


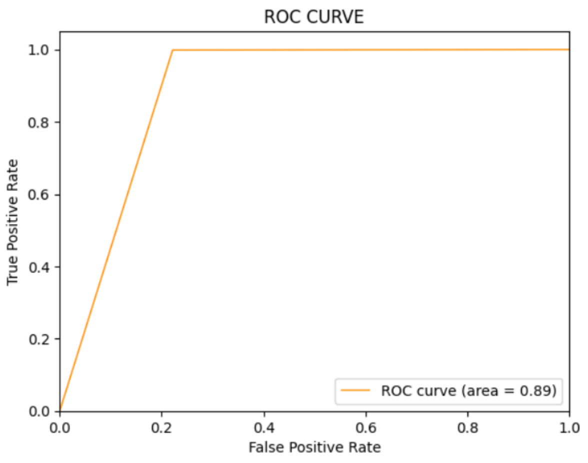
SUPPORT VECTOR MACHINE



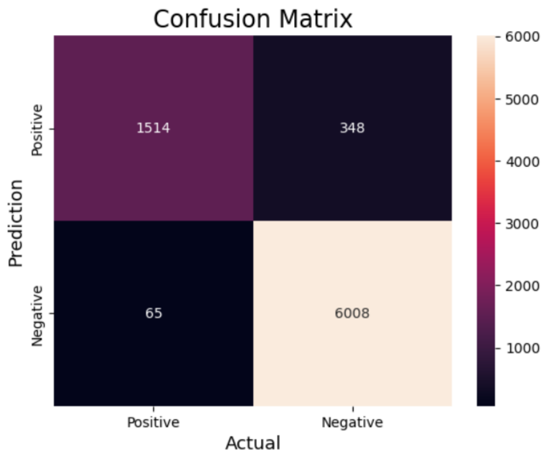


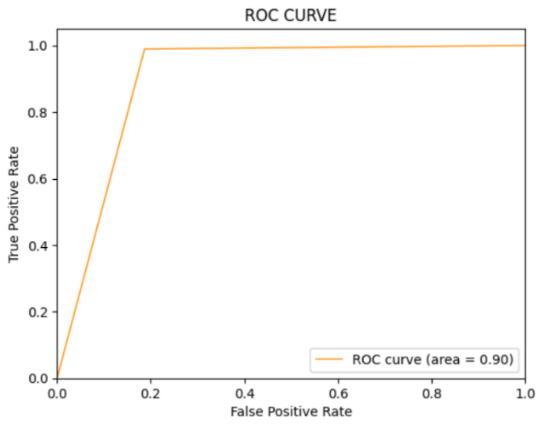
BERNOULLI NAÏVE BAYES



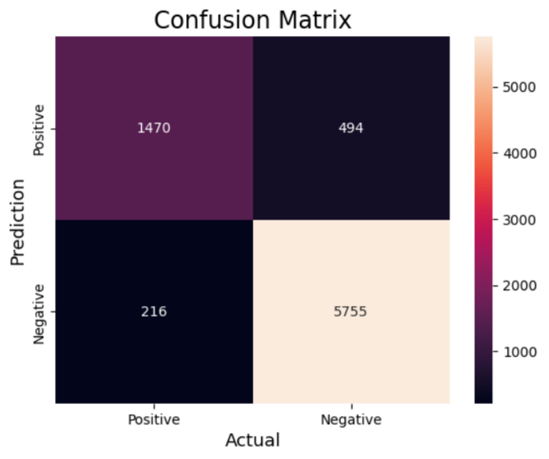


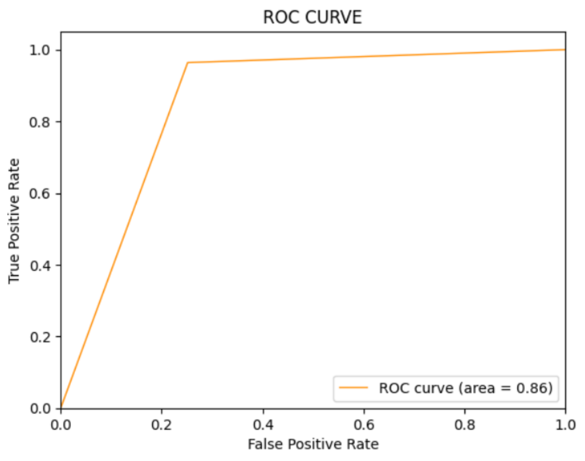
K NEAREST NEIGHBOR





LINEAR DISCRIMINANT ANALYSIS


Confusion Matrix
Prediction
Positive
Negative
1470
494
216
5755
Positive
Negative
Actual
5000
4000
3000
2000
1000


ROC CURVE
True Positive Rate
False Positive Rate
ROC curve (area = 0.86)
0.0
0.2
0.4
0.6
0.8
1.0

Chapter – 6

CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION

The project presents the findings and outcomes of the study based on the data collected and analyzed. This section aims to provide a comprehensive understanding of the performance of the machine learning model employed for Driver Fatigue Monitoring System and the insights derived from the analysis where the data is retrieved from Kaggle.

The datasets consist of Mouth Aspect ratio, Eye Aspect ratio, the distance between the points created on the eyes and mouth. The model's performance was assessed using multiple measures such as

accuracy, precision, recall, and F1-score. The analysis results show that the machine learning model did well in driver fatigue monitoring.

Various accuracies are achieved by different machine learning models as follows:

1) Logistic Regression: 94.16
2) Support Vector Machine: 94.35
3) Bernoulli Naïve Bayes: 94.27
4) Random Forest: 95.31
5) K Nearest Neighbor: 94.79
6) Linear Discriminant Analysis: 90.61

By above accuracies, though the difference is negligible, most suited classifier for the given data is Random Forest.

6.2 FUTURE SCOPE:

Driver fatigue monitoring systems (DFMS) can greatly increase road safety by identifying and warning drivers who are starting to feel sleepy. These systems can determine a driver's state of awareness by observing several cues, including eye movements, facial expressions, and signaling from the body.

Chapter – 7

REFERENCES

[1] Subbarao, A., Sahithya, K. (2019) Driver Drowsiness Detection System for Vehicle Safety, International Journal of Innovative Technology and Exploring Engineering (IJITEE).

[2] Z. E. Bowden and C. T. Ragsdale, "The truck driver scheduling problem with fatigue monitoring," Decision Support Syst., vol. 110, pp. 20–31, Jun. 2018.

[3] A. Mittal, K. Kumar, S. Dhamija, and M. Kaur, "Head movement based driver drowsiness detection: A review of state-of-art techniques," in Proc. IEEE Int. Conf. Eng. Technol. (ICETECH), Mar. 2016

[4]. Consensus Statement: Fatigue and accidents in transport operations. J. Sleep Res. 9, 395 (2000)

[5] Horne, J., Reyner, L.: Vehicle accidents related to sleep: a review. Occupational and Environmental Medicine 56, 289–294 (1999) 10

[6] Zhang, G.; Yau, K.K.; Zhang, X.; Li, Y. Traffic accidents involving fatigue driving and their extent of casualties. Accid. Anal. Prev. 2016.

[7] Chaudhuri, A.; Routray, A. Driver Fatigue Detection through Chaotic Entropy Analysis of Cortical Sources Obtained From Scalp EEG Signals. IEEE Trans. Intell. Transp. Syst. 2019.

[8] R. Bittner, K. Hána, L. Poušek, P. Smrka, P. Schreib and P. Vysoký, Detecting of Fatigue States of a Car Driver, ISMDA 2000, LNCS 1933, pp. 260–274, 2000

[9] Yu, J., Park, S., Lee, S., & Jeon, M. (2018). Driver drowsiness detection using condition-adaptive representation learning framework. IEEE transactions on intelligent transportation systems, 20(11), 4206-4218.

[10] W. Zhang, B. Cheng and Y. Lin, "Driver drowsiness recognition based on computer vision technology", Tsinghua Science and Technology, Vol. 17, No. 3, pp. 354-362, 2012.

[11] M. Singh, G. Kaur, "Drowsiness detection on eye blink Duration using algorithm", International Journal of Emerging Technology and Advanced

Engineering, Vol. 2, Issue 4, April 2012.

[12] Sukrit Mehta, Sharad Dadhich, Sahil Gumber, Arpita Jadhav Bhatt (2019). Real-Time Driver Drowsiness Detection System Using Eye Aspect Ratio and Eye Closure Ratio International Conference on Sustainable Computing in Science, Technology and Management.

[13] Tayab Khan, M., Anwar, H., Ullah, F., Ur Rehman, A., Ullah, R., Iqbal, A., … Kwak, K. S. (2019). Smart Real-Time Video Surveillance Platform for Drowsiness Detection Based on Eyelid Closure. Wireless Communications and Mobile Computing, 2019.

[14] RamalathaMarimuthu, A. Suresh, M. Alamelu and S.Kanagaraj "Driver fatigue detection using image processing and accident prevention", International journal of pure and applied mathematics, vol. 116,2017

[15] Assari, M. A., & Rahmati, M. (2011). Driver drowsiness detection using face expression recognition. 2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)