MACHINE LEARNING

(Distracted Driver Detection)

Summer Internship Report Submitted in partial

fulfillment of the requirement for undergraduate degree

Of

Bachelor of Technology

In

Computer Science and Engineering

By

P.VENKATA MADHUSUDHAN

221710307048

Under the Guidance of

____

Assistant Professor

GITAM (DEEMED TO BE UNIVERSITY)

VISAKHAPATNAM HYDERABAD BENGALURU

Department Of Computer Science and Engineering

GITAM School of Technology

GITAM (Deemed to be University) Hyderabad-502329

DECLARATION

I submitted this industrial training work entitled **“Distracted Driver Detection**” to GITAM (Deemed To Be University), Hyderabad in partial fulfillment of the requirements for the award of the degree of “**Bachelor of Technology**” in “**Computer Science and Engineering**”. I declare that it was carried out independently by me under the guidance of **Mr. XXXX**, Asst. Professor, GITAM (Deemed To Be University), Hyderabad, India.

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

Place: Hyderabad P.VENKATA
MADHUSUDHAN

Date: 21-07-2020 221710307048

GITAM (DEEMED TO BE UNIVERSITY)

Hyderabad-502329, India

Dated:21-7-2020

CERTIFICATE

This is to certify that the Industrial Training Report entitled **“Distracted Driver Detection”** is being submitted by P.Venkata Madhusudhan (221710307048) in partial fulfillment of the requirement for the award of **Bachelor of Technology in Computer Science And Engineering** at GITAM (Deemed to Be University), Hyderabad during the academic year 2019-20.

It is faithful record work carried out by her at the **Computer Science And Engineering Department**, GITAM University Hyderabad Campus under my guidance and supervision.

Dr. S.Phani Kumar

Assistant Professor

Professor and HOD

ACKNOWLEDGEMENT

Apart from my effort, the success of this internship largely depends on the encouragement and guidance of many others. I take this opportunity to express my gratitude to the people who have helped me in the successful competition of this internship.

I would like to thank, respected **Dr. N. Siva Prasad**, Pro Vice-Chancellor, GITAM Hyderabad, and Dr. CH. Sanjay, Principal, GITAM Hyderabad.

I would like to thank respected **Prof. S. Phani Kumar**, Head of the Department of Computer Science Engineering for giving me such a wonderful opportunity to expand my knowledge for my own branch and giving me guidelines to present the internship report. It helped me a lot to realize what we study for.

I would like to thank the respected faculties _____ who helped me to make this internship a successful accomplishment.

I would also like to thank my friends who helped me to make my work more organized and well-stacked till the end.

P.VENKATA MADHUSUDHAN

221710307048

ABSTRACT

In addition to vehicle control, drivers often perform secondary tasks that impede driving. Reduction of driver distraction is an important challenge for the safety of intelligent transportation systems. In this paper, a methodology for the detection and evaluation of driver distraction while performing secondary tasks is described and an appropriate hardware and a software environment is offered and studied. The system includes a model of normal driving, a subsystem for measuring the errors from the secondary tasks, and a module for total distraction evaluation. A new machine learning algorithm defines driver performance in lane keeping and speed maintenance on a specific road segment. To recognize the errors, a method is proposed, which compares normal driving parameters with ones obtained while conducting a secondary task. To evaluate distraction, an effective fuzzy logic algorithm is used. To verify the proposed approach, a case study with driver-in-the-loop experiments was carried out, in which participants performed the secondary task, namely chatting on a cell phone. The results presented in this research confirm its capability to detect and to precisely measure a level of abnormal driver performance.

Table of contents:

List of Figures used:

1. MACHINE LEARNING

1.1. Introduction:

Over the past two decades Machine Learning has become one of the mainstays of information technology and with that, a rather central, albeit usually hidden, part of our life. With the ever increasing amounts of data becoming available there is good reason to believe that smart data analysis will become even more pervasive as a necessary ingredient for technological progress.

Human designers often produce machines that do not work as well as desired in the environments in which they are used. In fact, certain characteristics of the working environment might not be completely known at design time. Machine learning methods can be used for on-the-job improvement of existing machine designs. The amount of knowledge available about certain tasks might be too large for explicit encoding by humans. Machines that learn this knowledge gradually might be able to capture more of it than humans would want to write down. Environments change over time. Machines that can adapt to a changing environment would reduce the need for constant redesign. New knowledge about tasks is constantly being discovered by humans. Vocabulary changes. There is a constant stream of new events in the world. Continuing redesign of AI systems to conform to new knowledge is impractical, but machine learning methods might be able to track much of it.

1.2. Importance of Machine Learning:

Machine learning is a branch of artificial intelligence that aims at enabling machines to perform their jobs skillfully by using intelligent software. The statistical learning methods constitute the backbone of intelligent software that is used to develop machine intelligence. Because machine learning algorithms require data to learn, the discipline must have connection with the discipline of databases. Similarly, there are familiar terms such as Knowledge Discovery from Data (KDD), data mining, and pattern recognition. One wonders how to view the big picture in which such connection is illustrated.

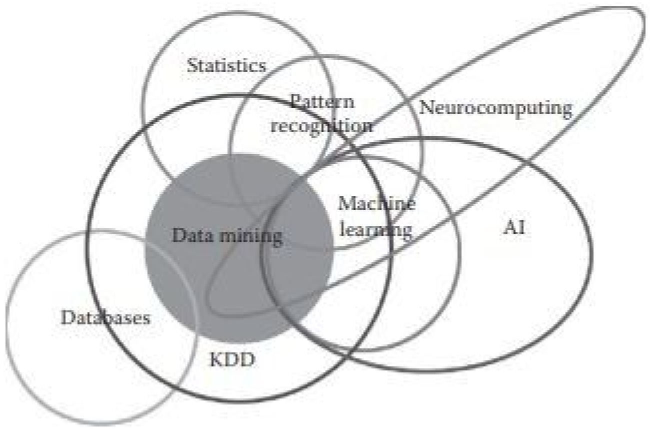


Fig 1.1 usage of Machine learning in different fields

There are some tasks that humans perform effortlessly or with some efforts, but we are unable to explain how we perform them. For example, we can recognize the speech of our friends without much difficulty. If we are asked how we recognize the voices, the answer is very difficult for us to explain. Because of the lack of understanding of such phenomenon (speech recognition in this case), we cannot craft algorithms for such scenarios. Machine learning algorithms are helpful in bridging this gap of understanding.

The idea is very simple. We are not targeting to understand the underlying processes that help us learn. We write computer programs that will make machines learn and enable them to perform tasks, such as prediction. The goal of learning is to construct a model that takes the input and produces the desired result. Sometimes, we can understand the model, whereas, at other times, it can also be like a black box for us, the working of which cannot be intuitively explained. The model can be considered as an approximation of the process we want machines to mimic. In such a situation, it is possible that we obtain errors for some input, but most of the time, the model provides correct answers. Hence, another measure of performance (besides performance of metrics of speed and memory usage) of a machine learning algorithm will be the accuracy of results.

1.3. Uses of Machine Learning:

Artificial Intelligence (AI) is everywhere. Possibility is that you are using it in one way or the other and you don't even know about it. One of the popular applications of AI is Machine Learning (ML), in which computers, software, and devices perform via cognition (very similar to the human brain). Herein, we share a few examples of machine learning that we use everyday and perhaps have no idea that they are driven by ML. These are some of the uses and applications of ML.

i. Virtual Personal Assistants:

Siri, Alexa, Google Now are some of the popular examples of virtual personal assistants. As the name suggests, they assist in finding information, when asked over voice. All you need to do is activate them and ask "What is my schedule for today?", "What are the flights from Germany to London", or similar questions. For answering, your personal assistant looks out for the information, recalls your related queries, or sends a command to other resources (like phone apps) to collect info. You can even instruct assistants for certain tasks like “Set an alarm for 6 AM next morning”, “Remind me to visit the Visa Office day after tomorrow”.

Machine learning is an important part of these personal assistants as they collect and refine the information on the basis of your previous involvement with them. Later, this set of data utilized to render results that are tailored to your preferences.

Virtual Assistants are integrated to a variety of platforms. For example:

- Smart Speakers: Amazon Echo and Google Home
- Smartphones: Samsung Bixby on Samsung S10
- Mobile Apps: Google Allo

ii. Predictions while commuting:

Traffic Predictions*:* We all have been using GPS navigation services. While we do that, our current locations and velocities are being saved at a central server for managing traffic. This data is then used to build a map of current traffic. While this helps in preventing the traffic and does congestion analysis, the underlying problem is that there are less number of cars that are equipped with GPS. Machine learning in such scenarios helps to estimate the regions where congestion can be found on the basis of daily experiences.

Online Transportation Networks*:* When booking a cab, the app estimates the price of the ride. When sharing these services, how do they minimize the detours? The answer is machine learning. Jeff Schneider, the engineering lead at Uber ATC reveals in an interview that they use

ML to define price surge hours by predicting the rider demand. In the entire cycle of the services, ML is playing a major role.

iii. Social Media Services:

From personalizing your news feed to better ads targeting, social media platforms are utilizing machine learning for their own and user benefits. Here are a few examples that you must be noticing, using, and loving in your social media accounts, without realizing that these wonderful features are nothing but the applications of ML.

- **People You May Know:** Machine learning works on a simple concept: understanding with experiences. Facebook continuously notices the friends that you connect with, the profiles that you visit very often, your interests, workplace, or a group that you share with someone etc. On the basis of continuous learning, a list of Facebook users are suggested that you can become friends with.
- **Face Recognition:** You upload a picture of you with a friend and Facebook instantly recognizes that friend. Facebook checks the poses and projections in the picture, notice the unique features, and then match them with the people in your friend list. The entire process at the backend is complicated and takes care of the precision factor but seems to be a simple application of ML at the front end.
- **Similar Pins:** Machine learning is the core element of Computer Vision, which is a technique to extract useful information from images and videos. Pinterest uses computer vision to identify the objects (or pins) in the images and recommends similar pins accordingly.

iv. Search Engine Result Refining:

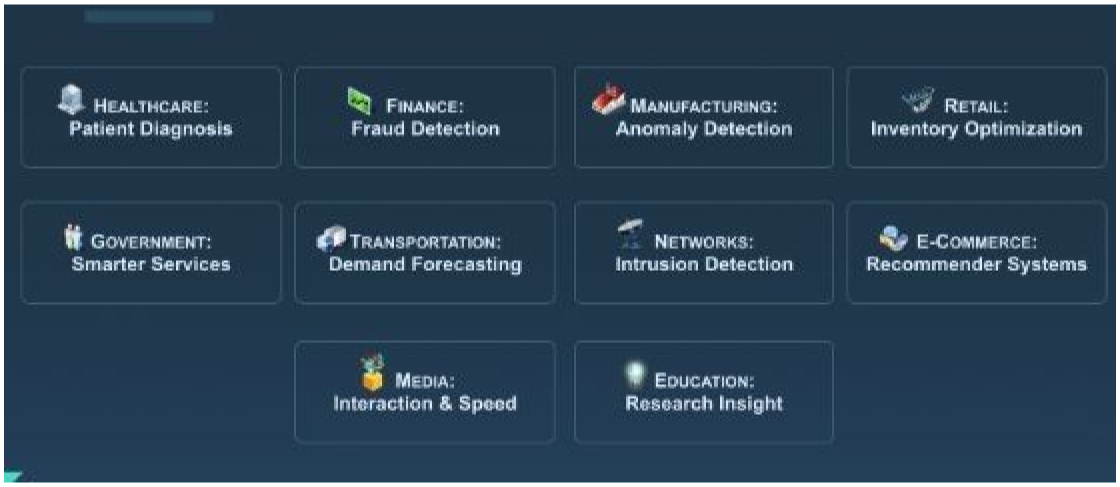
Google and other search engines use machine learning to improve the search results for you. Every time you execute a search, the algorithms at the backend keep a watch at how you respond to the results. If you open the top results and stay on the web page for long, the search engine assumes that the the results it displayed were in accordance to the query. Similarly, if you reach the second or third page of the search results but do not open any of the results, the search engine estimates that the results served did not match requirement. This way, the algorithms working at the backend improve the search results.

v. Product Recommendations:

You shopped for a product online few days back and then you keep receiving emails for shopping suggestions. If not this, then you might have noticed that the shopping website or the app recommends you some items that somehow matches with your taste. On the basis of your behaviour with the website/app, past purchases, items liked or added to cart, brand preferences etc., the product recommendations are made.

vi. Online Fraud Detection:

Machine learning is proving its potential to make cyberspace a secure place and tracking monetary frauds online is one of its examples. For example: Paypal is using ML for protection against money laundering. The company uses a set of tools that helps them to compare millions of transactions taking place and distinguish between legitimate or illegitimate transactions taking place between the buyers and sellers.

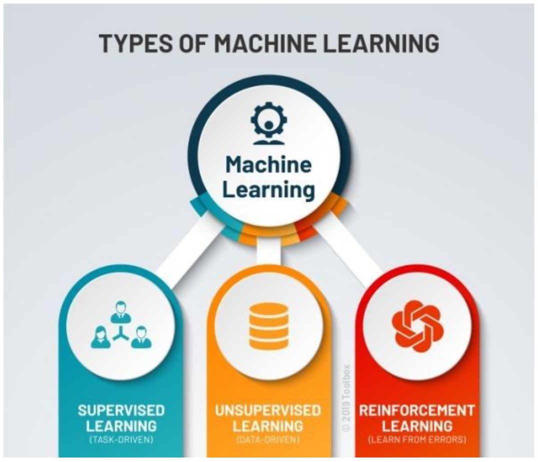


HEALTHCARE: Patient Diagnosis
FINANCE: Fraud Detection
MANUFACTURING: Anomaly Detection
RETAIL: Inventory Optimization
GOVERNMENT: Smarter Services
TRANSPORTATION: Demand Forecasting
NETWORKS: Intrusion Detection
E-COMMERCE: Recommender Systems
MEDIA: Interaction & Speed
EDUCATION: Research Insight

Fig 1.2 Uses of Machine learning

1.4. Types of Machine Learning:

There are 3 types of Machine learning which are widely used in today's world these are:



TYPES OF MACHINE LEARNING
Machine Learning
SUPERVISED LEARNING (TASK-DRIVEN)
UNSUPERVISED LEARNING (DATA-DRIVEN)
REINFORCEMENT LEARNING (LEARN FROM ERRORS)

Fig 1.3 types of ML

1.4.1 Supervised Learning:

Supervised learning is one of the most basic types of machine learning. In this type, the machine learning algorithm is trained on labeled data. Even though the data needs to be labeled accurately for this method to work, supervised learning is extremely powerful when used in the right circumstances.In supervised learning, the ML algorithm is given a small training dataset to work with. This training dataset is a smaller part of the bigger dataset and serves to give the algorithm a basic idea of the problem, solution, and data points to be dealt with. The training dataset is also very similar to the final dataset in its characteristics and provides the algorithm with the labeled parameters required for the problem. The algorithm then finds relationships between the parameters given, essentially establishing a cause and effect relationship between the variables in the dataset. At the end of the training, the algorithm has an idea of how the data works and the relationship between the input and the output.

1.4.2 Unsupervised Learning:

Unsupervised machine learning holds the advantage of being able to work with unlabeled data. This means that human labor is not required to make the dataset machine-readable, allowing much larger datasets to be worked on by the program.In supervised learning, the labels allow the algorithm to find the exact nature of the relationship between any two data points. However, unsupervised learning does not have labels to work off of, resulting in the creation of hidden structures. Relationships between data points are perceived by the algorithm in an abstract manner, with no input required from human beings.The creation of these hidden structures is what makes unsupervised learning algorithms versatile. Instead of a defined and set problem statement, unsupervised learning algorithms can adapt to the data by dynamically changing hidden structures. This offers more post-deployment development than supervised learning algorithms.

1.4.3 Reinforcement Learning:

It directly takes inspiration from how human beings learn from data in their lives. It features an algorithm that improves upon itself and learns from new situations using a trial-and-error method. Favorable outputs are encouraged or 'reinforced', and non-favorable outputs are discouraged or 'punished'.Based on the psychological concept of conditioning, reinforcement learning works by putting the algorithm in a work environment with an interpreter and a reward system. In every iteration of the algorithm, the output result is given to the interpreter, which decides whether the outcome is favorable or not.In case of the program finding the correct solution, the interpreter reinforces the solution by providing a reward to the algorithm. If the outcome is not favorable, the algorithm is forced to reiterate until it finds a better result. In most cases, the reward system is directly tied to the effectiveness of the result. In typical reinforcement learning use-cases, such as finding the shortest route between two points on a map, the solution is not an absolute value. Instead, it takes on a score of effectiveness, expressed in a percentage value. The higher this percentage value is, the more reward is given to the algorithm. Thus, the program is trained to give the best possible solution for the best possible reward.

2. DEEP LEARNING

2.1. Deep Learning Importance:

Deep learning algorithms run data through several “layers” of neural network algorithms, each of which passes a simplified representation of the data to the next layer.Most machine learning algorithms work well on datasets that have up to a few hundred features, or columns.

Basically deep learning is itself a subset of machine learning but in this case the machine learns in a way in which humans are supposed to learn. The structure of deep learning models is highly similar to a human brain with a large number of neurons and nodes like neurons in the human brain thus resulting in an artificial neural network. In applying traditional machine learning algorithms we have to manually select input features from complex data set and then train them which becomes a very tedious job for ML scientist but in neural networks we don’t have to manually select useful input features, there are various layers of neural networks for handling complexity of the data set and algorithm as well. In my recent project on human activity recognition , when we applied traditional machine learning algorithm like K-NN then we have to separately detect human and its activity also had to select impactful input parameters manually which became a very tedious task as data set was way too complex but the complexity dramatically reduced on applying artificial neural network, such is the power of deep learning. Yes it’s correct that deep learning algorithms take lots of time for training sometimes even weeks as well but its execution on new data is so fast that it's not even comparable with traditional ML algorithms. Deep learning has enabled Industrial Experts to overcome challenges which were impossible a decade ago like Speech and Image recognition and Natural Language Processing. Majority of the Industries are currently depending on it , be it Journalism, Entertainment, Online Retail Store, Automobile, Banking and Finance, Healthcare, Manufacturing or even Digital Sector. Video recommendations, Mail Services, Self Driving cars, Intelligent Chat bots, Voice Assistants are just trending achievements of Deep Learning.

Furthermore, Deep learning can most profoundly be considered as the future of Artificial Intelligence due to constant rapid increase in amount of data as well as the gradual development in the hardware field as well, resulting in better computational power.

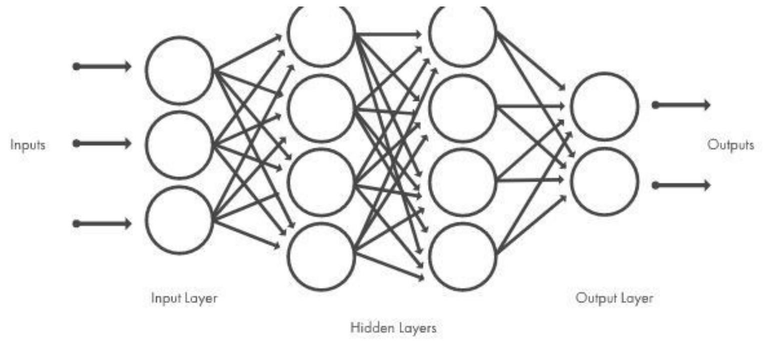


Fig 2.1 Deep Neural network

2.2. Uses of Deep Learning:

i. Translations:

Although automatic machine translation isn’t new, deep learning is helping enhance automatic translation of text by using stacked networks of neural networks and allowing translations from images.

ii. Adding color to black-and-white images and videos:

It used to be a very time-consuming process where humans had to add color to black-and-white images and videos by hand can now be automatically done with deep-learning models.

iii. Language recognition:

Deep learning machines are beginning to differentiate dialects of a language. A machine decides that someone is speaking English and then engages an AI that is learning to tell the differences between dialects. Once the dialect is determined, another AI will step in that specializes in that particular dialect. All of this happens without involvement from a human.

iv. Autonomous vehicles:

There's not just one AI model at work as an autonomous vehicle drives down the street. Some deep-learning models specialize in streets signs while others are trained to recognize pedestrians. As a car navigates down the road, it can be informed by up to millions of individual AI models that allow the car to act.

v. Computer vision:

Deep learning has delivered super-human accuracy for image classification, object detection, image restoration and image segmentation—even handwritten digits can be recognized. Deep learning using enormous neural networks is teaching machines to automate the tasks performed by human visual systems.

vi.Text generation:

The machines learn the punctuation, grammar and style of a piece of text and can use the model it developed to automatically create entirely new text with the proper spelling, grammar and style of the example text. Everything from Shakespeare to Wikipedia entries have been created.

vii. Deep-learning robots:

Deep-learning applications for robots are plentiful and powerful from an impressive deep-learning system that can teach a robot just by observing the actions of a human completing a task to a housekeeping robot that's provided with input from several other AIs in order to take action. Just like how a human brain processes input from past experiences, current input from senses and any additional data that is provided, deep-learning models will help robots execute tasks based on the input of many different AI opinions.

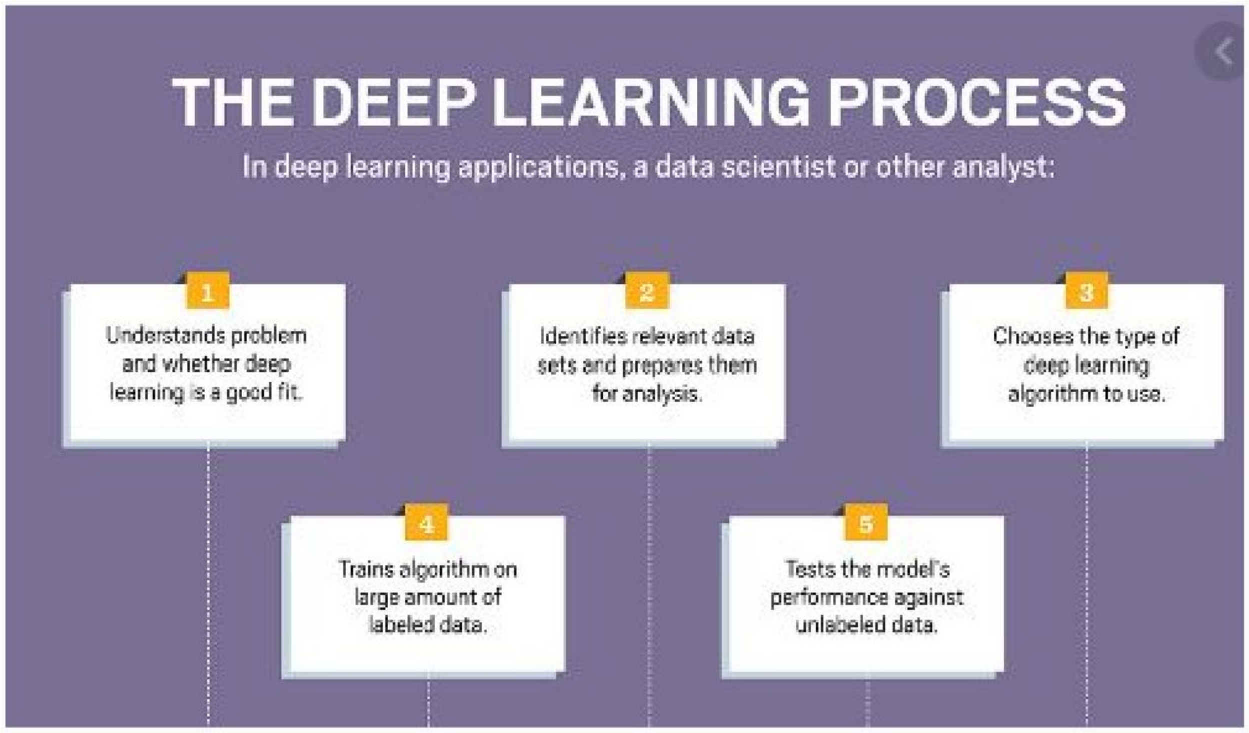

THE DEEP LEARNING PROCESS
In deep learning applications, a data scientist or other analyst:
1
Understands problem and whether deep learning is a good fit.
2
Identifies relevant data sets and prepares them for analysis.
3
Chooses the type of deep learning algorithm to use.
4
Trains algorithm on large amount of labeled data.
5
Tests the model's performance against unlabeled data.

Fig 2.2 The deep learning process

2.3. Relation between Data Mining, Machine Learning and Deep Learning:

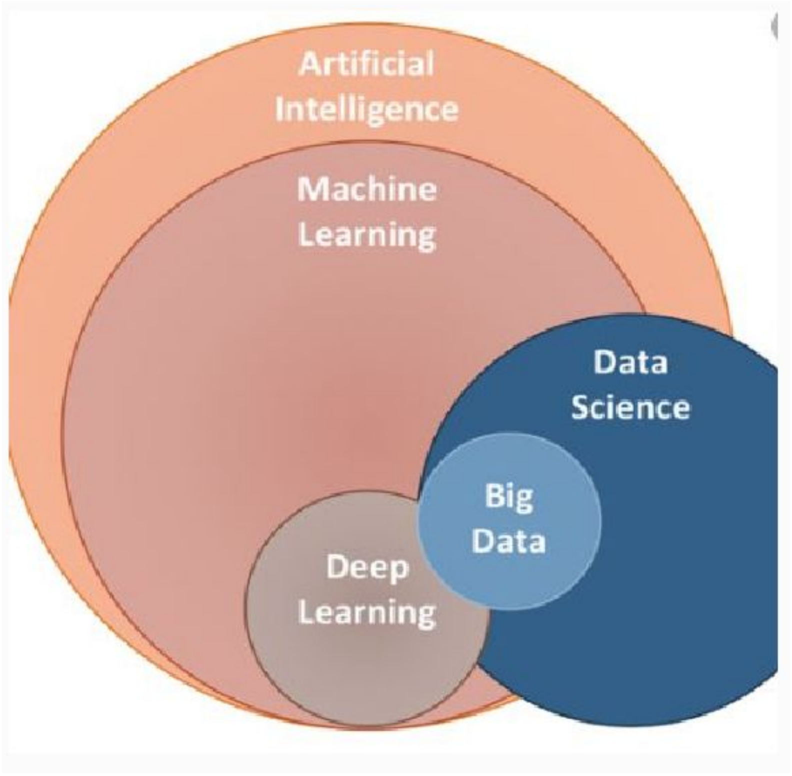


Fig 2.3 Relation between DM,ML,DL

The deep learning, data mining and machine learning share a foundation in data science, and there certainly is overlap between the two. Data mining can use machine learning algorithms to improve the accuracy and depth of analysis, and vice-versa; machine learning can use mined data as its foundation, refining the dataset to achieve better results.

You could also argue that data mining and machine learning are similar in that they both seek to address the question of how we can learn from data. However, the way in which they achieve this end, and their applications, form the basis of some significant differences.

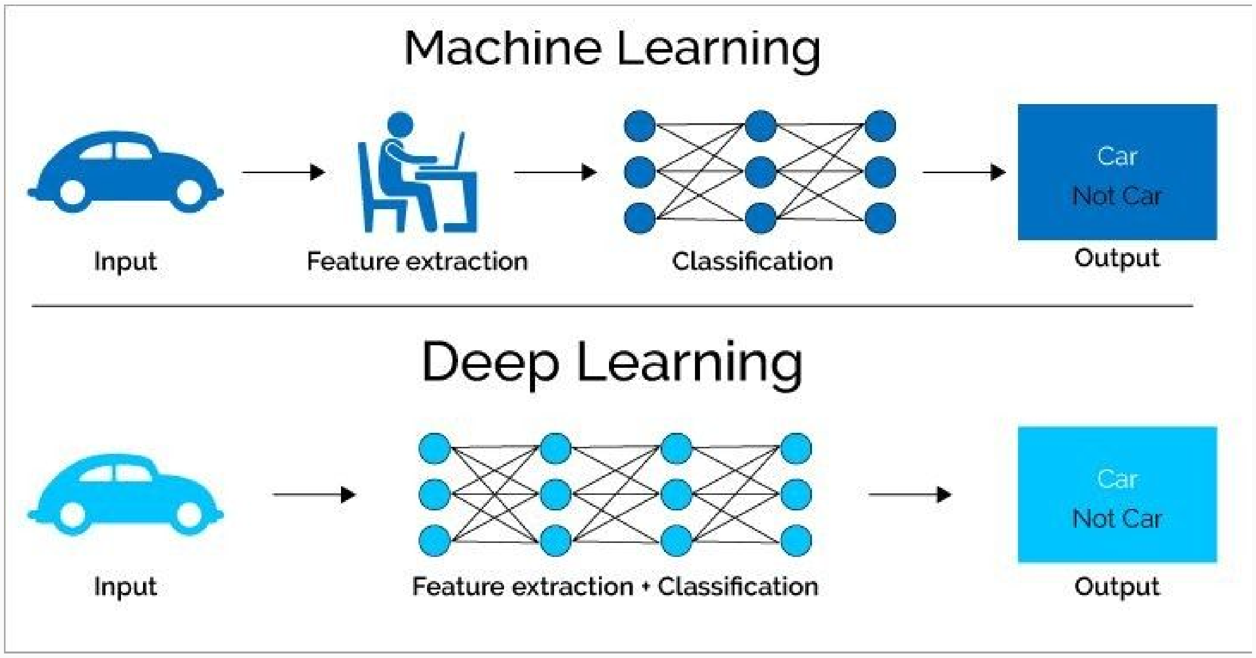


Fig 2.4 process in machine learning and deep learning

Machine Learning comprises of the ability of the machine to learn from trained data set and predict the outcome automatically. It is a subset of artificial intelligence.

Deep Learning is a subset of machine learning. It works in the same way on the machine just like how the human brain processes information. Like a brain can identify the patterns by comparing it with previously memorized patterns, deep learning also uses this concept.

Deep learning can automatically find out the attributes from raw data while machine learning selects these features manually which further needs processing. It also employs artificial neural networks with many hidden layers, big data, and high computer resources.

Data Mining is a process of discovering hidden patterns and rules from the existing data. It uses relatively simple rules such as association, correlation rules for the decision-making process, etc. Deep Learning is used for complex problem processing such as voice recognition etc. It uses Artificial Neural Networks with many hidden layers for processing. At times data mining also uses deep learning algorithms for processing the data.

3. PYTHON

3.1. Introduction:

Python is a widely used general-purpose, high level programming language. It was created by Guido van Rossum in 1991 and further developed by the Python Software Foundation. It was designed with an emphasis on code readability, and its syntax allows programmers to express their concepts in fewer lines of code.Python is a programming language that lets you work quickly and integrate systems more efficiently.

Python is dynamically typed and garbage-collected.It supports multiple programming paradigms,including structured , object-oriented,and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

3.2. Setup of Python:

- Python distribution is available for a wide variety of platforms. You need to download only the binary code applicable for your platform and install Python.
- The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python https://www.python.org/

3.2.1 Installation(using python IDLE):

- To start, go to python.org/downloads and then click on the button to download the latest version of Python.
- We can download python IDLE in windows,mac and linux operating systems also.

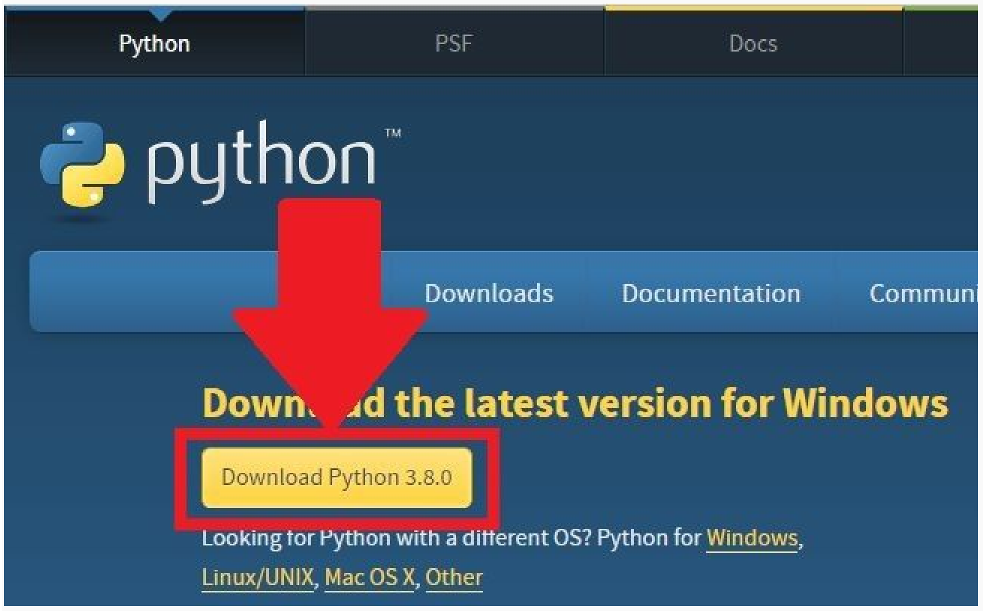
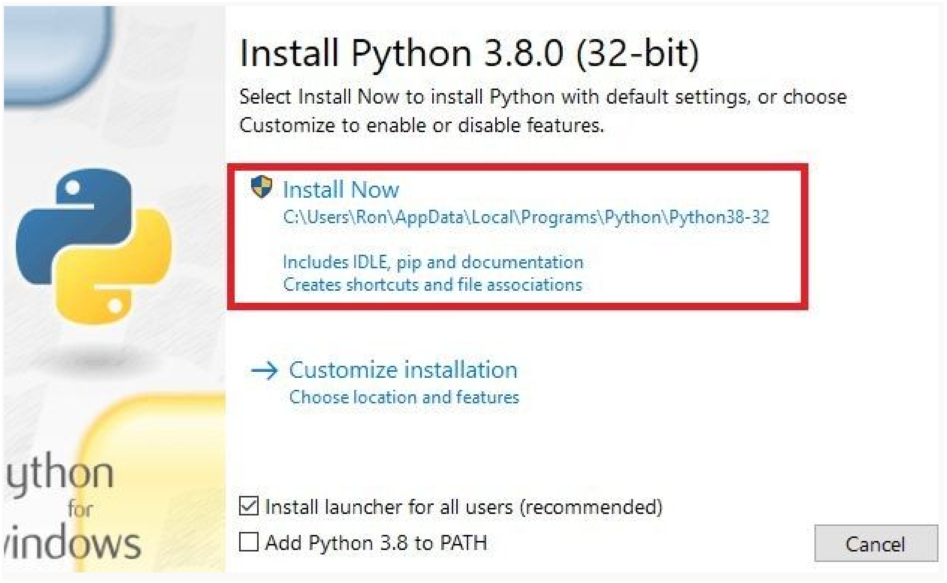

Python
PSF
Docs
python™
Downloads
Documentation
Communi
Download the latest version for Windows
Download Python 3.8.0
Looking for Python with a different OS? Python for Windows,
Linux/UNIX, Mac OS X, Other

Figure 3.2.1 : Python download

- Run the .exe file that you just downloaded and start the installation of Python by clicking on Install Now.
- We can give environmental variable i.e path after completion of downloading



Install Python 3.8.0 (32-bit)

Select Install Now to install Python with default settings, or choose Customize to enable or disable features.

Install Now
C:\Users\Ron\AppData\Local\Programs\Python\Python38-32

Includes IDLE, pip and documentation
Creates shortcuts and file associations

→ Customize installation
Choose location and features

☑ Install launcher for all users (recommended)
☐ Add Python 3.8 to PATH

Fig 3.2.1.1 python installation

- When python is installed, a program called IDLE is also installed along with it.It provides a graphical user interface to work with python.

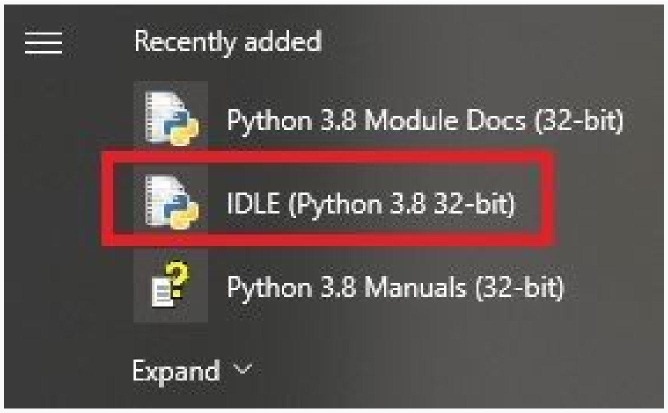
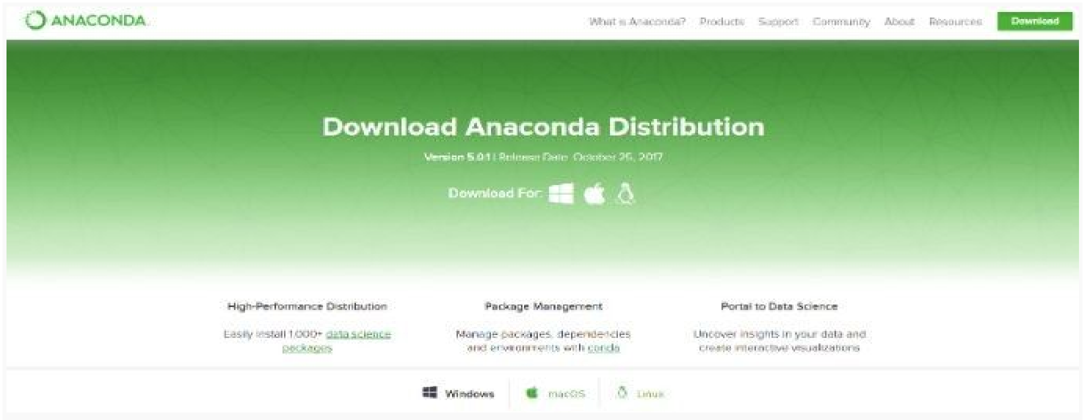


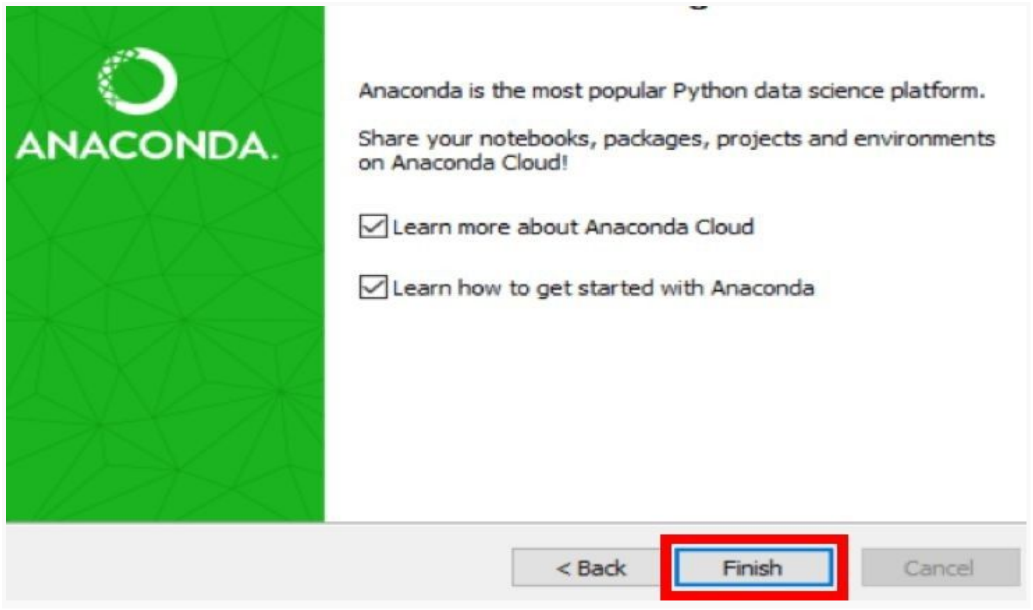
Fig 3.2.1.2 IDLE

3.2.2. Python Installation using Anaconda:

- Anaconda is a free open source distribution of python for large scale data processing, predictive analytics and scientific computing.
- Conda is a package manager quickly installs and manages packages. Anaconda for Windows installation:
  1. Go to the following link: Anaconda.com/downloads


ANACONDA
What is Anaconda? Products Support Community About Resources Download
Download Anaconda Distribution
Version 5.0.1 | Release Date: October 25, 2017
Download For:
High-Performance Distribution
Easily install 1,000+ data science packages
Package Management
Manage packages, dependencies and environments with conda
Portal to Data Science
Uncover insights in your data and create interactive visualizations
Windows
macOS
Linux

2. Download python 3.4 version for (32-bitgraphic installer/64 -bit graphic installer)
3. Select path(i.e. add anaconda to path & register anaconda as default python 3.4)
4. Click finish



Anaconda is the most popular Python data science platform.

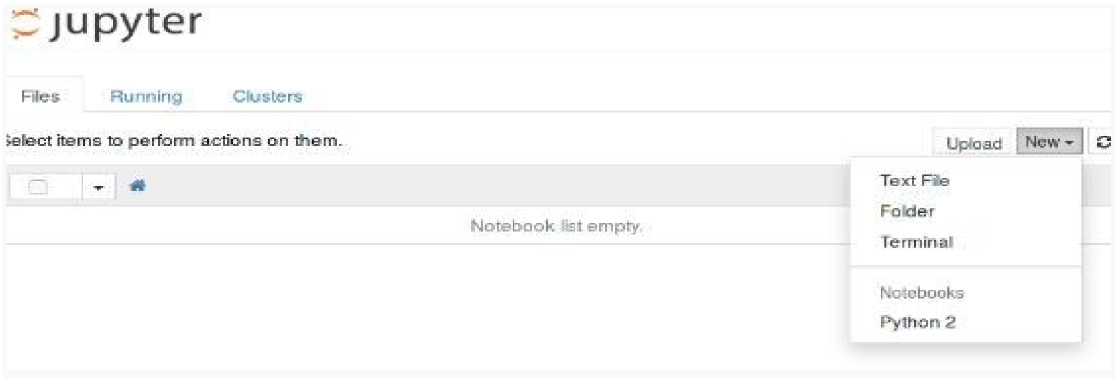
Share your notebooks, packages, projects and environments on Anaconda Cloud!

[✓] Learn more about Anaconda Cloud

[✓] Learn how to get started with Anaconda

Fig 3.2.2.1 After installation

5. Open jupyter notebook



Select items to perform actions on them.

Notebook list empty.

Fig 3.2.2.2 jupyter notebook

3.3. Features:

1. **Readable:** Python is a very readable language.
2. **Easy to Learn:** Learning python is easy as this is a expressive and high level programming language, which means it is easy to understand the language and thus easy to learn.
3. **Cross platform:** Python is available and can run on various operating systems such as Mac, Windows, Linux, Unix etc. This makes it a cross platform and portable language.
4. **Open Source:** Python is an open source programming language.
5. **Large standard library:** Python comes with a large standard library that has some handy codes and functions which we can use while writing code in Python.
6. **Free:** Python is free to download and use. This means you can download it for free and use it in your application. Python is an example of a FLOSS (Free/Libre Open Source Software), which means you can freely distribute copies of this software, read its source code and modify it.
7. **Supports exception handling:** If you are new, you may wonder what is an exception? An exception is an event that can occur during program exception and can disrupt the normal flow of program. Python supports exception handling which means we can write less error prone code and can test various scenarios that can cause an exception later on.
8. **Advanced features:** Supports generators and list comprehensions. We will cover these features later.
9. **Automatic memory management:** Python supports automatic memory management which means the memory is cleared and freed automatically. You do not have to bother clearing the memory.

3.4. Variable Types:

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory. Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals or characters in these variables.

Python has five standard data types –

- Numbers
- Strings
- Lists
- Tuples
- Dictionary

3.4.1 Python Numbers:

Number data types store numeric values. They are immutable data types, means that changing the value of a number data type results in a newly allocated object.

Python supports four different numerical types −

- **int (signed integers)** − They are often called just integers or ints, are positive or negative whole numbers with no decimal point.
- **long (long integers)** − Also called longs, they are integers of unlimited size, written like integers and followed by an uppercase or lowercase L.
- **float (floating point real values)** − Also called floats, they represent real numbers and are written with a decimal point dividing the integer and fractional parts. Floats may also be in scientific notation, with E or e indicating the power of 10(2.5e2=2.5x10^2=250).

3.4.2. Python Strings:

In Python, Strings can be created by simply enclosing characters in quotes. Python does not support character types. These are treated as length-one strings, and are also considered as substrings. Substrings are immutable and can’t be changed once created.Strings are the ordered blocks of text that are enclosed in single or double quotations. Thus, whatever is written in quotes, is considered as string. Though it can be written in single or double quotations, double quotation marks allow the user to extend strings over multiple lines without backslashes, which is usually the signal of continuation of an expression, e.g., ‘abc’, “ABC”.

3.4.3. Python Lists:

- List is a collection data type in python. It is ordered and allows duplicate entries as well. Lists in python need not be homogeneous, which means it can contain different data types like integers, strings and other collection data types. It is mutable in nature and allows indexing to access the members in a list.
- To declare a list, we use the square brackets.
- List is like any other array that we declare in other programming languages. Lists in python are often used to implement stacks and queues. The lists are mutable in nature. Therefore, the values can be changed even after a list is declared.

3.4.4 Python Tuples:

- A tuple is a collection of objects which ordered and immutable. Tuples are sequences, just like lists. The differences between tuples and lists are, the tuples cannot be changed unlike lists and tuples use parentheses, whereas lists use square brackets.Creating a tuple is as simple as putting different comma-separated values. Optionally you can put these comma-separated values between parentheses also.

3.4.5 Python Dictionary:

- It is a collection data type just like a list or a set, but there are certain features that make python dictionary unique. A dictionary in python is not ordered and is changeable as well. We can make changes in a dictionary unlike sets or strings which are immutable in nature. Dictionary contains key-value pairs like a map that we have in other programming languages. A dictionary has indexes. Since the value of the keys we declare in a dictionary are always unique, we can use them as indexes to access the elements in a dictionary.

3.5. Functions:

3.5.1. Defining a Function:

- Function blocks begin with the keyword def followed by the function name and parentheses (( )).
- Any input parameters or arguments should be placed within these parentheses. You can also define parameters inside these parentheses.
- The first statement of a function can be an optional statement - the documentation string of the function or docstring.
- The code block within every function starts with a colon (:) and is indented.
- The statement return [expression] exits a function, optionally passing back an expression to the caller. A return statement with no arguments is the same as return None.

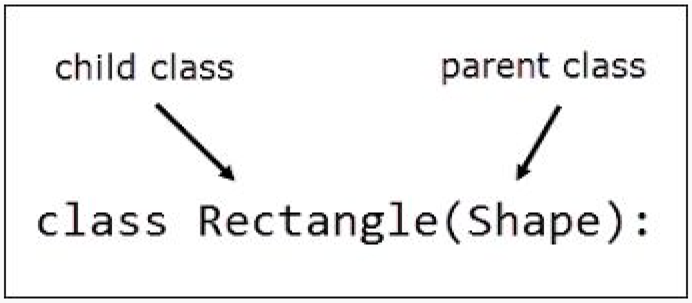
3.5.2 Calling a Function:

- Defining a function only gives it a name, specifies the parameters that are to be included in the function and structures the blocks of code.
- Once the basic structure of a function is finalized, you can execute it by calling it from another function or directly from the Python prompt.

3.6 OOPs Concepts:

3.6.1 Class:

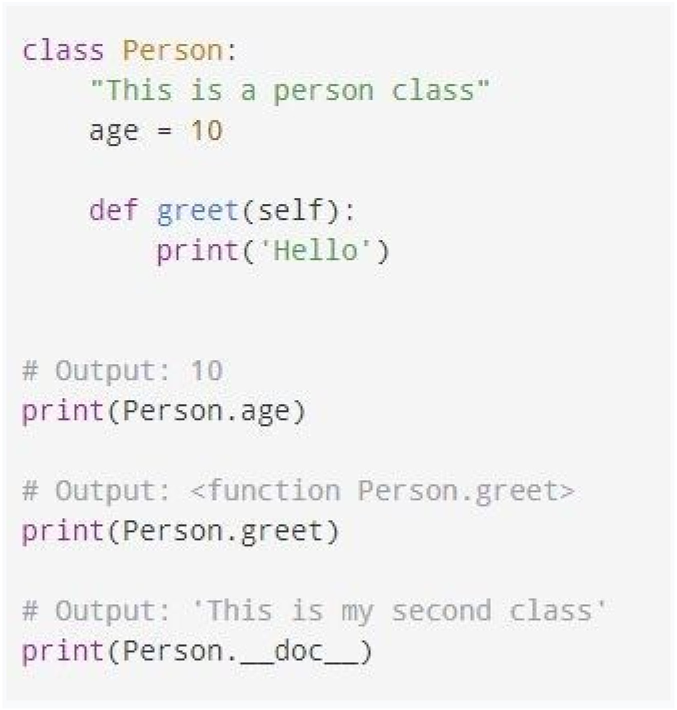
- Python is an object oriented programming language. Unlike procedure oriented programming, where the main emphasis is on functions, object oriented programming stresses on objects.
- An object is simply a collection of data (variables) and methods (functions) that act on those data. Similarly, a class is a blueprint for that object.
- We can think of class as a sketch (prototype) of a house. It contains all the details about the floors, doors, windows etc. Based on these descriptions we build the house. House is the object.
- As many houses can be made from a house's blueprint, we can create many objects from a class. An object is also called an instance of a class and the process of creating this object is calledinstantiation.
- Like function definitions begin with the def keyword in Python, class definitions begin with a class keyword.
- The first string inside the class is called docstring and has a brief description about the class.



```
class Rectangle(Shape):
```

Fig 3.6.1 Class defining

- As soon as we define a class, a new class object is created with the same name. This class object allows us to access the different attributes as well as to instantiate new objects of that class.



```python
class Person:
    "This is a person class"
    age = 10

    def greet(self):
        print('Hello')


# Output: 10
print(Person.age)

# Output: <function Person.greet>
print(Person.greet)

# Output: 'This is my second class'
print(Person.__doc__)
```

Fig 3.6.1.1 Example of class

4. DISTRACTED DRIVER DETECTION

4.1. Project Requirements:

4.1.1 Packages Used:

- **Numpy:** In Python we have lists that serve the purpose of arrays, but they are slow to process.NumPy aims to provide an array object that is up to 50x faster that traditional Python lists.The array object in NumPy is called ndata, it provides a lot of supporting functions that make working with ndarray very easy.Arrays are very frequently used in data science, where speed and resources are very important.
- **Pandas:**Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.
- **Seaborn:** Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

Matplotlib: Matplotlib is one of the most popular Python packages used for data visualization. It is a cross-platform library for making 2D plots from data in arrays. Matplotlib is written in Python and makes use of NumPy, the numerical mathematics extension of Python. It provides an object-oriented API that helps in embedding plots in applications using Python GUI toolkits such as PyQt, WxPythonotTkinter. It can be used in Python and IPython shells, Jupyter notebook and web application servers also.Matplotlib has a procedural interface named the Pylab, which is designed to resemble MATLAB, a proprietary programming language developed by MathWorks. Matplotlib along with NumPy can be considered as the open source equivalent of MATLAB.

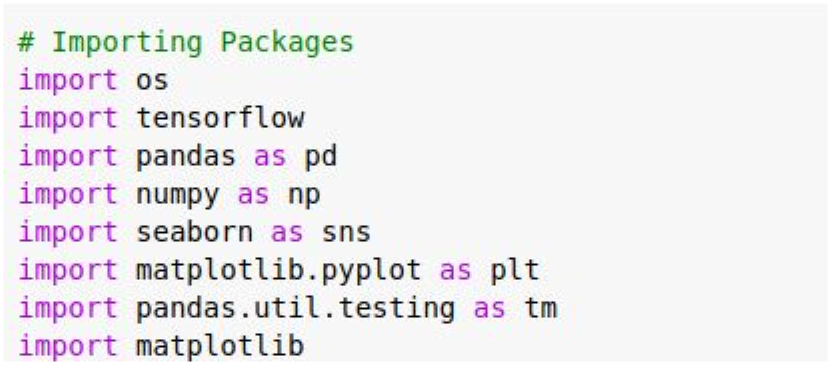
Tensorflow: TensorFlow is Google Brain's second-generation system. Version 1.0.0 was released on February 11, 2017.[10] While the reference implementation runs on single devices, TensorFlow can run on multiple CPUs and GPUs (with optional CUDA and SYCL extensions for general-purpose computing on graphics processing units).[11] TensorFlow is available on 64-bit Linux, macOS, Windows, and mobile computing platforms including Android and iOS.

Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices.

TensorFlow computations are expressed as stateful dataflow graphs. The name TensorFlow derives from the operations that such neural networks perform on multidimensional data arrays, which are referred to as *tensors*. During the Google I/O Conference in June 2016, Jeff Dean stated that 1,500 repositories on GitHub mentioned TensorFlow, of which only 5 were from Google.[

In March 2018, Google announced TensorFlow.js version 1.0 for machine learning in JavaScript.[13]

In May 2019, Google announced TensorFlow Graphics for deep learning in computer graphics.

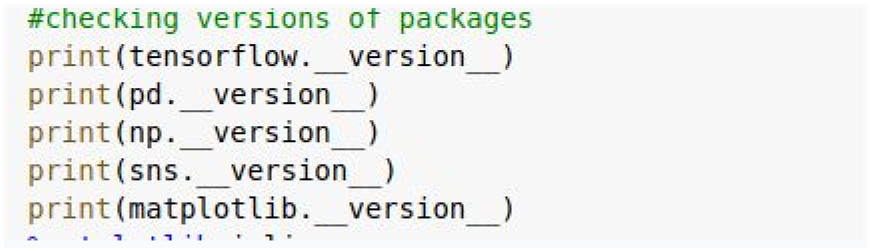


```
# Importing Packages
import os
import tensorflow
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import pandas.util.testing as tm
import matplotlib
```

Fig 4.1.1 packages

4.1.2. Versions of Packages:

The versions of the packages are found by following command



```
#checking versions of packages
print(tensorflow.__version__)
print(pd.__version__)
print(np.__version__)
print(sns.__version__)
print(matplotlib.__version__)
```

Fig 4.1.2 versions

4.1.3 Algorithms Used:

Here algorithm used is:

- Convolutional Neural Network(CNN)

4.2. Problem Statement:

Statement: We've all been there: a light turns green and the car in front of you doesn't budge. Or, a previously unremarkable vehicle suddenly slows and starts swerving from side-to-side.

When you pass the offending driver, what do you expect to see? You certainly aren't surprised when you spot a driver who is texting, seemingly enraptured by social media, or in a lively hand-held conversation on their phone.

According to the CDC motor vehicle safety division, one in five car accidents is caused by a distracted driver. Sadly, this translates to 425,000 people injured and 3,000 people killed by distracted driving every year.

State Farm hopes to improve these alarming statistics, and better insure their customers, by testing whether dashboard cameras can automatically detect drivers engaging in distracted behaviors. Given a dataset of 2D dashboard camera images, State Farm is challenging Kagglers to classify each driver's behavior. Are they driving attentively, wearing their seatbelt, or taking a selfie with their friends in the backseat?

4.3. Dataset Description:

In this project, the images of various drivers in their driving positions have been captured and collected. And, the data set thus obtained is classified into two sets - train set, and validation set with a split percentage of 70% for train data set and 30% for validation data set.

The obtained each train and validation datasets have been classified into two classes, namely, C0 and C1 for safe-driving and texting-right respectively. The C0 folder represents a safe driving position of the driver and the C1 folder represents an unsafe/distracted driving position.

4.4. Objective of the Case Study:

Driving a car is a complex task, and it requires complete attention. Distracted driving is any activity that takes away the driver's attention from the road. Several studies have identified three main types of distraction: visual distractions (driver's eyes off the road), manual distractions (driver's hands off the wheel) and cognitive distractions (driver's mind off the driving task).

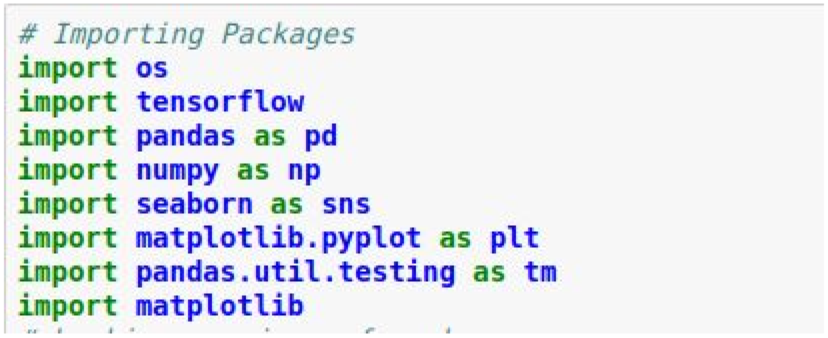
The National Highway Traffic Safety Administration (NHTSA) reported that 36,750 people died in motor vehicle crashes in 2018, and 12% of it was due to distracted driving. Texting is the most alarming distraction. Sending or reading a text takes your eyes off the road for 5 seconds. At 55 mph, that’s like driving the length of an entire football field with your eyes closed.

Many states now have laws against texting, talking on a cell phone, and other distractions while driving. We believe that computer vision can augment the efforts of the governments to prevent accidents caused by distracted driving. Our algorithm automatically detects the distracted activity of the drivers and alerts them. We envision this type of product being embedded in cars to prevent accidents due to distracted driving.

5. DATA PREPROCESSING/FEATURE ENGINEERING AND EDA

5.1. Importing the packages:

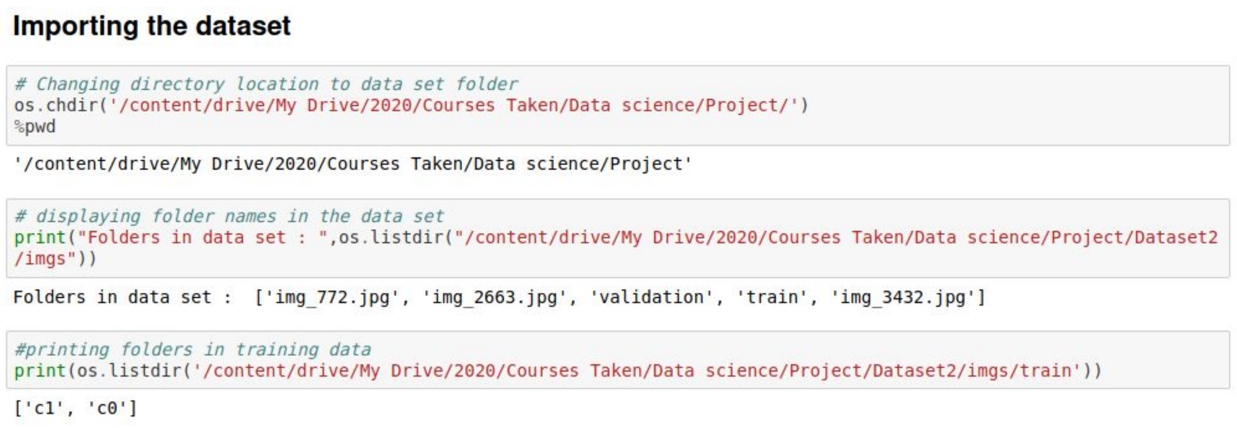
Import the necessary packages required to perform Data pre processing steps



```
# Importing Packages
import os
import tensorflow
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import pandas.util.testing as tm
import matplotlib
```

Fig 5.1importing packages

5.2. Importing Dataset:



Importing the dataset

```
# Changing directory location to data set folder
os.chdir('/content/drive/My Drive/2020/Courses Taken/Data science/Project/')
%pwd
```

```
'/content/drive/My Drive/2020/Courses Taken/Data science/Project'
```

```
# displaying folder names in the data set
print("Folders in data set : ",os.listdir("/content/drive/My Drive/2020/Courses Taken/Data science/Project/Dataset2
/imgs"))
```

```
Folders in data set :  ['img_772.jpg', 'img_2663.jpg', 'validation', 'train', 'img_3432.jpg']
```

```
#printing folders in training data
print(os.listdir('/content/drive/My Drive/2020/Courses Taken/Data science/Project/Dataset2/imgs/train'))
```

```
['c1', 'c0']
```

Fig 5.2 data set

5.3. Generating Plots:

5.3.1: Visualize the data of all the Features

Here we have displayed the count of each feature in training and testing data using bar plots

Count of training images

```
#visualizing counts of features
length0 = len(os.listdir('/content/drive/My Drive/2020/Courses Taken/Data science/Project/Dataset2/imgs/train/c0'))
length1 = len(os.listdir('/content/drive/My Drive/2020/Courses Taken/Data science/Project/Dataset2/imgs/train/c1'))
sns.barplot([length0,length1],['c0','c1'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb54b4279e8>
```

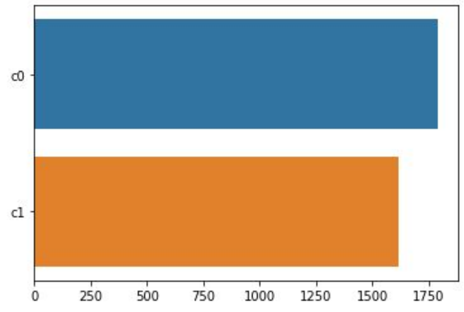


Fig 5.3.1 count of training images

Count of validation images

```
#visualizing counts of features
length2 = len(os.listdir('/content/drive/My Drive/2020/Courses Taken/Data science/Project/Dataset2/imgs/validation/
c0'))
length3 = len(os.listdir('/content/drive/My Drive/2020/Courses Taken/Data science/Project/Dataset2/imgs/validation/
c1'))
sns.barplot([length2,length3],['c0','c1'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fb4fe6cbe10>
```

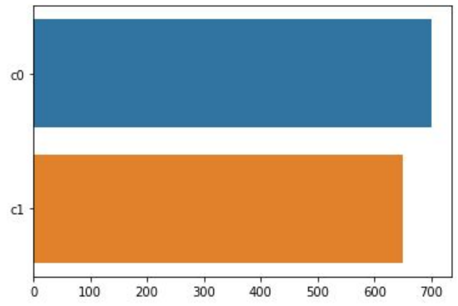
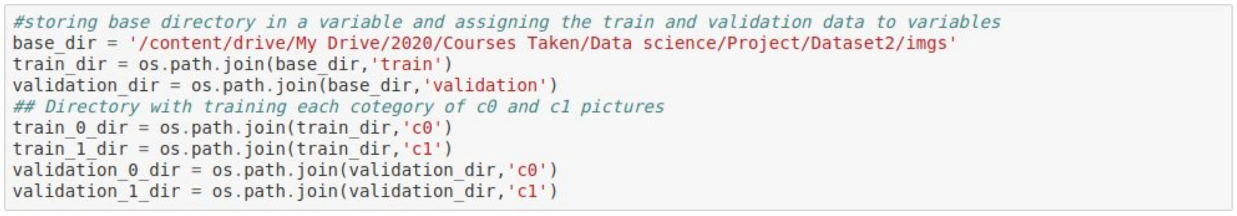


Fig 5.3.2 count of testing images

6. FEATURE SELECTION

6.1. Train-Validation-Split:

One of the first decisions to make when starting a modeling project is how to utilize the existing data. One common technique is to split the data into two groups typically referred to as the training and testing sets. The training set is used to develop models and feature sets; they are the substrate for estimating parameters, comparing models, and all of the other activities required to reach a final model. The test set is used only at the conclusion of these activities for estimating a final, unbiased assessment of the model's performance. It is critical that the test set not be used prior to this point. Looking at the test sets results would bias the outcomes since the testing data will have become part of the model development process.

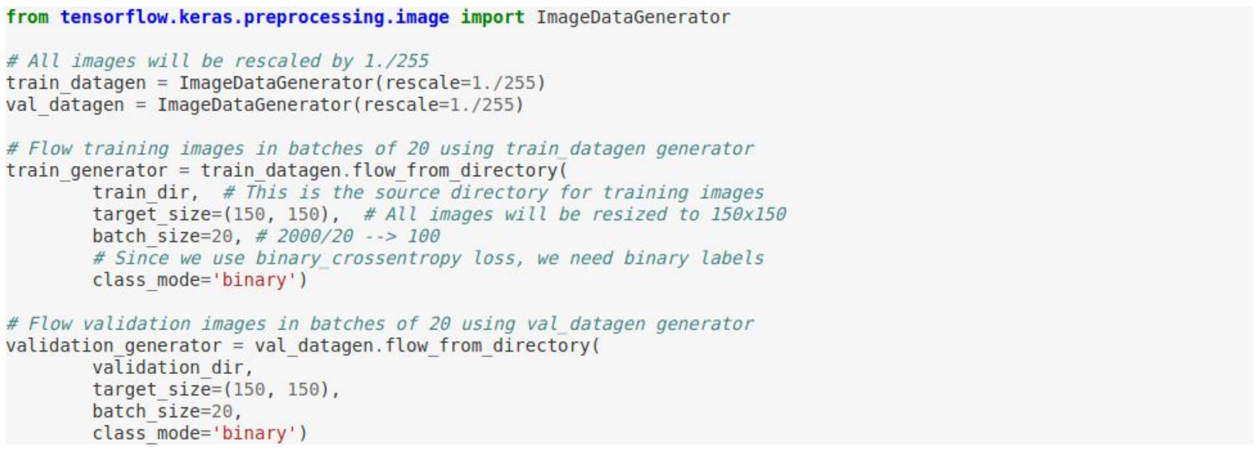


```
#storing base directory in a variable and assigning the train and validation data to variables
base_dir = '/content/drive/My Drive/2020/Courses Taken/Data science/Project/Dataset2/imgs'
train_dir = os.path.join(base_dir,'train')
validation_dir = os.path.join(base_dir,'validation')
## Directory with training each cotegory of c0 and c1 pictures
train_0_dir = os.path.join(train_dir,'c0')
train_1_dir = os.path.join(train_dir,'c1')
validation_0_dir = os.path.join(validation_dir,'c0')
validation_1_dir = os.path.join(validation_dir,'c1')
```

Fig 6.1 Train-Validation-Split

6.2. Image-Scaling:

It is a step of Data Pre Processing which is applied to independent variables or features of data. It basically helps to normalise the data within a particular range. Sometimes, it also helps in speeding up the calculations in an algorithm. Real world dataset contains features that highly vary in magnitudes, units, and range. Normalisation should be performed when the scale of a feature is irrelevant or misleading and not should Normalise when the scale is meaningful.



```python
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# All images will be rescaled by 1./255
train_datagen = ImageDataGenerator(rescale=1./255)
val_datagen = ImageDataGenerator(rescale=1./255)

# Flow training images in batches of 20 using train_datagen generator
train_generator = train_datagen.flow_from_directory(
        train_dir,  # This is the source directory for training images
        target_size=(150, 150),  # All images will be resized to 150x150
        batch_size=20, # 2000/20 --> 100
        # Since we use binary_crossentropy loss, we need binary labels
        class_mode='binary')

# Flow validation images in batches of 20 using val_datagen generator
validation_generator = val_datagen.flow_from_directory(
        validation_dir,
        target_size=(150, 150),
        batch_size=20,
        class_mode='binary')
```

Fig 6.2 Image Scaling

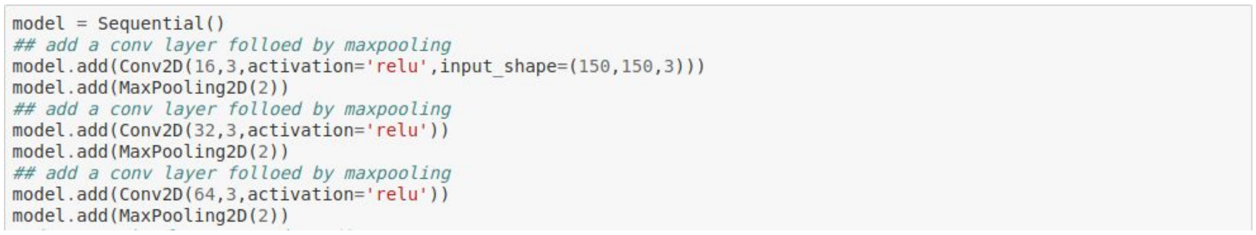
7. MODEL BUILDING AND EVALUATION

7.1. Brief about the Algorithms used:

Convolutional Neural Network (CNN):

Image classification is the task of taking an input image and outputting a class or a probability of classes that best describes the image. In CNN, we take an image as an input, assign importance to its various aspects/features in the image and be able to differentiate one from another. The preprocessing required in CNN is much lesser as compared to other classification algorithms.

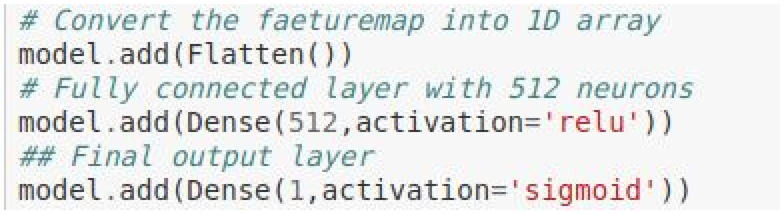
A CNN typically has three layers: a convolutional layer, pooling layer, and fully connected layer.Convolutional layer : The main objective of convolution is to extract features such as edges, colours, corners from the input. As we go deeper inside the network, the network starts identifying more complex features such as shapes,digits, face parts as well.



```
model = Sequential()
## add a conv layer folloed by maxpooling
model.add(Conv2D(16,3,activation='relu',input_shape=(150,150,3)))
model.add(MaxPooling2D(2))
## add a conv layer folloed by maxpooling
model.add(Conv2D(32,3,activation='relu'))
model.add(MaxPooling2D(2))
## add a conv layer folloed by maxpooling
model.add(Conv2D(64,3,activation='relu'))
model.add(MaxPooling2D(2))
```

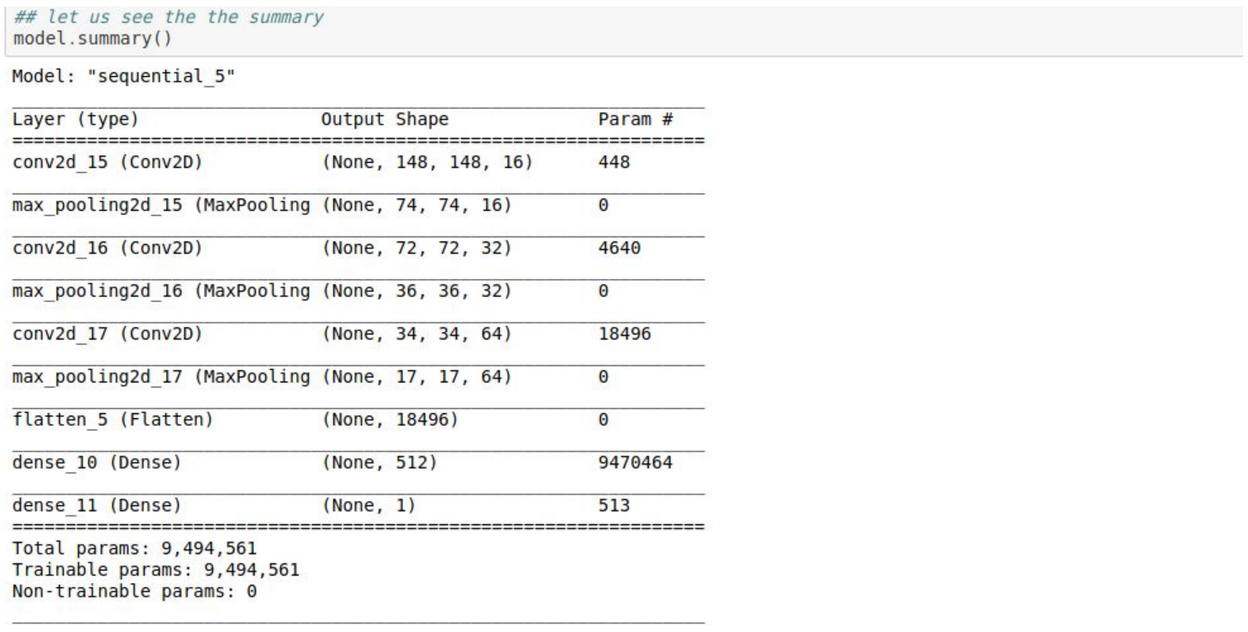
Fig 7.1.1 Convolutional layer

Add Dense layers on top : now that we have converted our input image into a suitable form for our Multi-Level fully connected architecture, we shall flatten the image into one column vector. The flattened output is fed to a feed-forward neural network and backpropagation applied to every iteration of training. Over a series of epochs, the model can distinguish between dominating and certain low-level features in images and classify them.



```
# Convert the faeturemap into 1D array
model.add(Flatten())
# Fully connected layer with 512 neurons
model.add(Dense(512,activation='relu'))
## Final output layer
model.add(Dense(1,activation='sigmoid'))
```

Fig 7.1.2 Desne layer



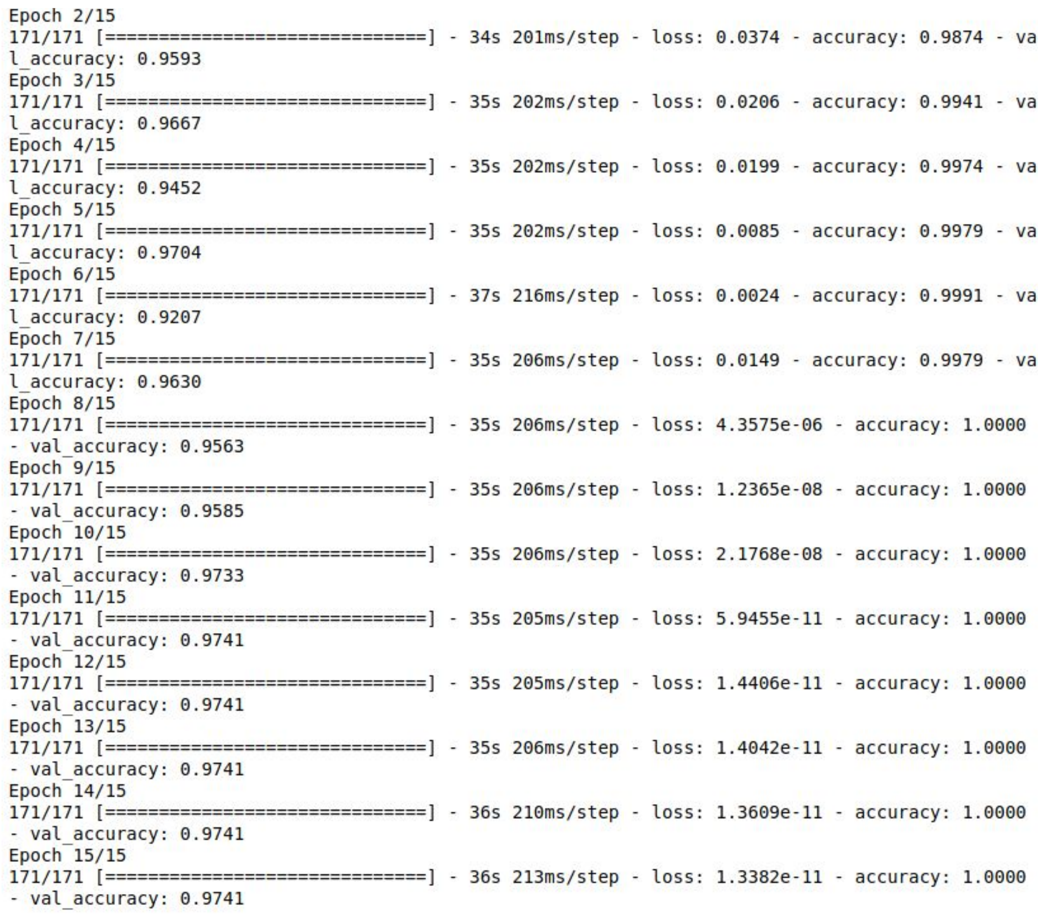
```
## let us see the the summary
model.summary()
```

```
Model: "sequential_5"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_15 (Conv2D)           (None, 148, 148, 16)      448       
_________________________________________________________________
max_pooling2d_15 (MaxPooling (None, 74, 74, 16)        0         
_________________________________________________________________
conv2d_16 (Conv2D)           (None, 72, 72, 32)        4640      
_________________________________________________________________
max_pooling2d_16 (MaxPooling (None, 36, 36, 32)        0         
_________________________________________________________________
conv2d_17 (Conv2D)           (None, 34, 34, 64)        18496     
_________________________________________________________________
max_pooling2d_17 (MaxPooling (None, 17, 17, 64)        0         
_________________________________________________________________
flatten_5 (Flatten)          (None, 18496)             0         
_________________________________________________________________
dense_10 (Dense)             (None, 512)               9470464   
_________________________________________________________________
dense_11 (Dense)             (None, 1)                 513       
=================================================================
Total params: 9,494,561
Trainable params: 9,494,561
Non-trainable params: 0
_________________________________________________________________
```

Fig 7.1.3 Model summary

7.2 Train the models:

Fitting on the Training set and making predictions on the Validation set



```
Epoch 2/15
171/171 [==============================] - 34s 201ms/step - loss: 0.0374 - accuracy: 0.9874 - va
l_accuracy: 0.9593
Epoch 3/15
171/171 [==============================] - 35s 202ms/step - loss: 0.0206 - accuracy: 0.9941 - va
l_accuracy: 0.9667
Epoch 4/15
171/171 [==============================] - 35s 202ms/step - loss: 0.0199 - accuracy: 0.9974 - va
l_accuracy: 0.9452
Epoch 5/15
171/171 [==============================] - 35s 202ms/step - loss: 0.0085 - accuracy: 0.9979 - va
l_accuracy: 0.9704
Epoch 6/15
171/171 [==============================] - 37s 216ms/step - loss: 0.0024 - accuracy: 0.9991 - va
l_accuracy: 0.9207
Epoch 7/15
171/171 [==============================] - 35s 206ms/step - loss: 0.0149 - accuracy: 0.9979 - va
l_accuracy: 0.9630
Epoch 8/15
171/171 [==============================] - 35s 206ms/step - loss: 4.3575e-06 - accuracy: 1.0000
- val_accuracy: 0.9563
Epoch 9/15
171/171 [==============================] - 35s 206ms/step - loss: 1.2365e-08 - accuracy: 1.0000
- val_accuracy: 0.9585
Epoch 10/15
171/171 [==============================] - 35s 206ms/step - loss: 2.1768e-08 - accuracy: 1.0000
- val_accuracy: 0.9733
Epoch 11/15
171/171 [==============================] - 35s 205ms/step - loss: 5.9455e-11 - accuracy: 1.0000
- val_accuracy: 0.9741
Epoch 12/15
171/171 [==============================] - 35s 205ms/step - loss: 1.4406e-11 - accuracy: 1.0000
- val_accuracy: 0.9741
Epoch 13/15
171/171 [==============================] - 35s 206ms/step - loss: 1.4042e-11 - accuracy: 1.0000
- val_accuracy: 0.9741
Epoch 14/15
171/171 [==============================] - 36s 210ms/step - loss: 1.3609e-11 - accuracy: 1.0000
- val_accuracy: 0.9741
Epoch 15/15
171/171 [==============================] - 36s 213ms/step - loss: 1.3382e-11 - accuracy: 1.0000
- val_accuracy: 0.9741
```

Fig 7.2 Compile and train the model

7.3 Validate the models:

To validate the model we have taken the accuracy and validation accuracy from the trained model and plotted them using legend() model of plotting.

```
train_acc = history.history['accuracy']
val_acc = history.history['val_accuracy']
train_loss = history.history['loss']
val_loss = history.history['val_loss']
epochs = list(range(1,16))
plt.subplot(2,1,1)
plt.plot(epochs,train_acc,label='train_acc')
plt.plot(epochs,val_acc,label='val_acc')
plt.title('accuracy')
plt.legend()
plt.subplot(2,1,2)
plt.plot(epochs,train_loss,label='train_loss')
plt.plot(epochs,val_loss,label='val_loss')
plt.title('loss')
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7f4fd0db7908>
```

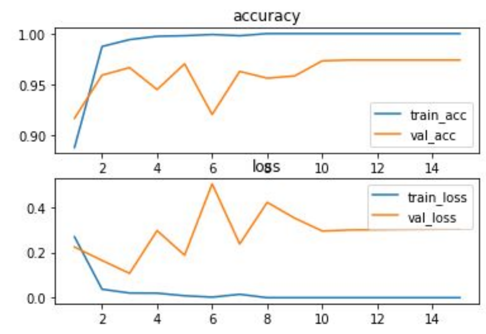


Fig 7.3 Validate the model

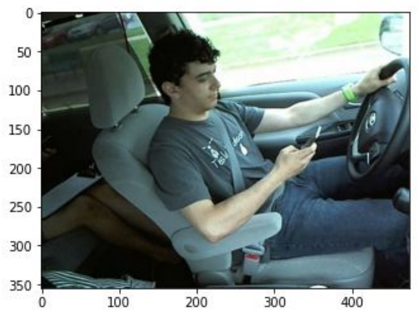
7.4 Predictions from raw data:

Here we have taken an image which the training model has not seen and not trained, the array size 1 indicates that the driver is distracted .

Testing with unseen data

```python
from tensorflow.keras.preprocessing import image
import numpy as np
img = image.load_img('/content/drive/My Drive/2020/Courses Taken/Data science/Project/external-content.duckduckgo.c
om.jpeg')
print(type(img))
plt.imshow(plt.imread('/content/drive/My Drive/2020/Courses Taken/Data science/Project/external-content.duckduckgo.
com.jpeg'))
img = tf.keras.preprocessing.image.img_to_array(img)
print(img.shape)
print(type(img))
img = tf.image.resize(img,(150,150))
## Scaling
img = img/255
print(img.shape)
img = np.expand_dims(img,axis=0)
print(img.shape)
```

```
<class 'PIL.JpegImagePlugin.JpegImageFile'>
(355, 474, 3)
<class 'numpy.ndarray'>
(150, 150, 3)
(1, 150, 150, 3)
```



```python
print(model.predict(img))
```

```
array([[1.]], dtype=float32)
```

Fig 7.4 Predicting with raw data

8. CONCLUSION

It is concluded after performing thorough Exploratory Data analysis which include statistical models which are computed to get accuracy which are computed to get a clear understanding of the data set and it is come to point of getting the solution for the problem statement being that the driver is distracted from driving.

From our experiments upon the given data using machine learning model Convolutional Neural Network we have found out that it has given a high yield of accuracy. Hence, it can be concluded that the same algorithm is best suited for our data model.

9. REFERENCES

[1] Machine learning

[2]Supervised Machine Learning: Model Validation, a Step by Step Approach

[3]https://www.tensorflow.org/tutorials/images/cnn