MACHINE LEARNING

(Object Detection)

Summer Internship Report Submitted in partial

fulfillment of the requirement for undergraduate

degree of

Bachelor of Technology

In

COMPUTER SCIENCE AND ENGINEERING

By

Motupally Sundara charya

221710302038

Under the Guidance of

Assistant Professor

Department Of Computer Science and Engineering
GITAM School of Technology

GITAM (Deemed to be University)

Hyderabad-502329

DECLARATION

I submit this industrial training work entitled **“Object 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.**, 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	Name: Motupally sundara charya
Date:	Student Roll No: 221710302001

GITAM (DEEMED TO BE UNIVERSITY)

Hyderabad-502329, India

Dated:

CERTIFICATE

This is to certify that the Industrial Training Report entitled **“Object Detection”** is being submitted by Motupally sundara charya(221710302038) 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 him 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. N.Seetharamaiah,** Principal, GITAM Hyderabad

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

I would like to thank the respected faculties **Mr.** 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.

Motupally Sundara charya

221710302038

ABSTRACT

Computer vision is the branch of the science of computers and software systems which can recognize as well as understand Images and scenes . Computer vision is consists of various aspects such as Image recognition , Object Detection ,Image generation , Image super-resolution and many more . Object detection is widely used for face detection ,vehicle detection , Pedestrian counting   , web images ,   security systems and self driving cars . In this project , we are using highly accurate object detection algorithms and methods such as R-CNN, Fast-RCNN, Faster-RCNN , and fast yet highly accurate ones like SSD and YOLO . Using these methods and algorithms ,based on dep learning which is also based on machine learning requires lots of mathematical and deep learning framework understanding by using dependencies such as Tensor Flow , OpenCV , imageai etc , we can detect each and every object in an image by the area object in an highlighted rectangular boxes and identify each and every object and assign its tag to the object . This also includes the accuracy of each method for identifying objects

Table of Contents

List of Figures

1. MACHINE LEARNING

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.

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 database. 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.

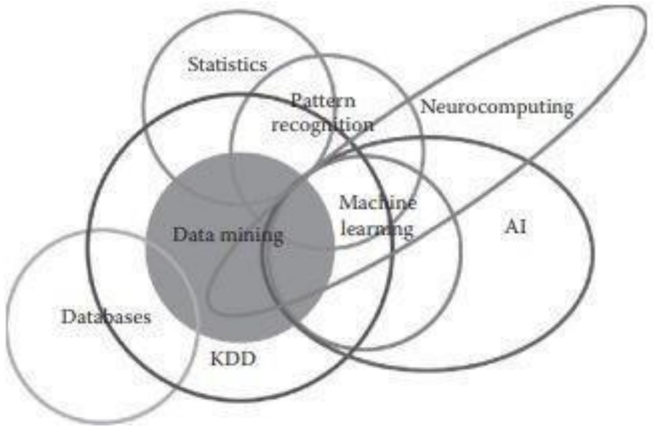


Fig 1.2 usage of Machine learning in different fields

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.

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 human brain). Herein, we share few examples of machine learning that we use everyday and perhaps have no idea that they are driven by ML. These are some 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 send 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 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 S8
- 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 a 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 recommend 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.

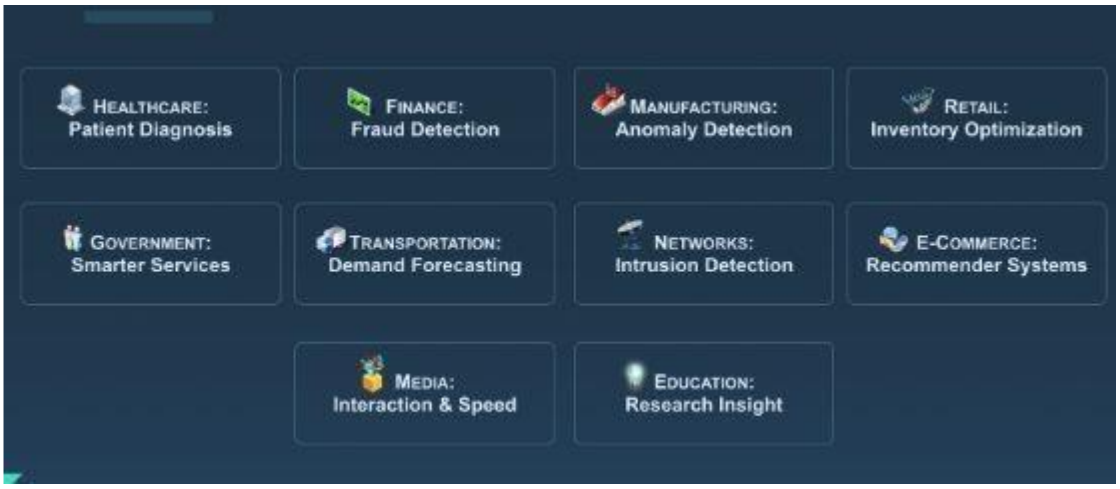

HEALTHCARE: Patient Diagnosis
FINANCE: Fraud Detection
MANUFACTURING: Anomaly Detection
RETAIL: Inventory Optimization
GOVERNMENT: Smarter Services
TRANSPORTATION: Demand Forecasting
NETWORKS: Intrusion Detection
E-COMMERCE: Recommender Systems
MEDIA: Interaction & Speed
EDUCATION: Research Insight

Fig 1.3 Uses of Machine learning

Types of Machine Learning:

There are 3 types of Machine learning which are widely used in todays world these Are:

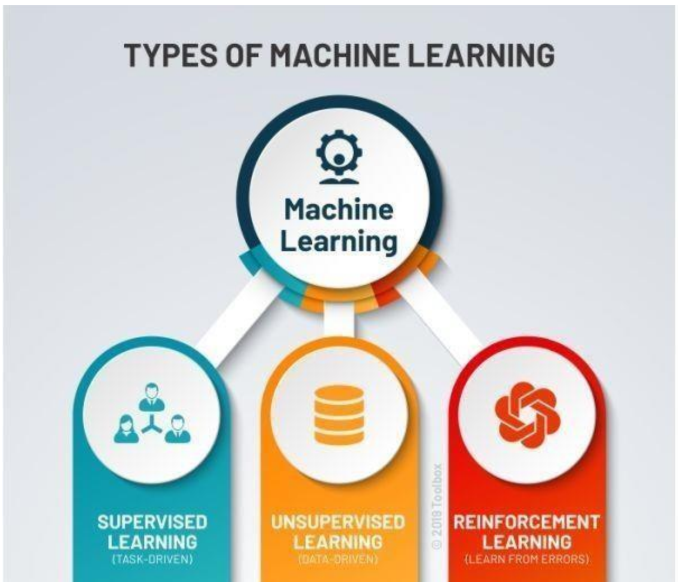

TYPES OF MACHINE LEARNING
Machine Learning
SUPERVISED LEARNING (TASK-DRIVEN)
UNSUPERVISED LEARNING (DATA-DRIVEN)
REINFORCEMENT LEARNING (LEARN FROM ERRORS)
© 2019 Toolbox

Fig 1.4 types of ML

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.

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.

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

Deep Learning and It's 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 model is highly similar to a human brain with large number of neurons and nodes like neurons in human brain thus resulting in 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 its not even comparable with traditional ML algorithms. Deep learning has enabled Industrial Experts to overcome challenges which were impossible, a decades 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 future of Artificial Intelligence due to constant rapid increase in amount of data as well as the gradual development in hardware field as well, resulting in better computational power.

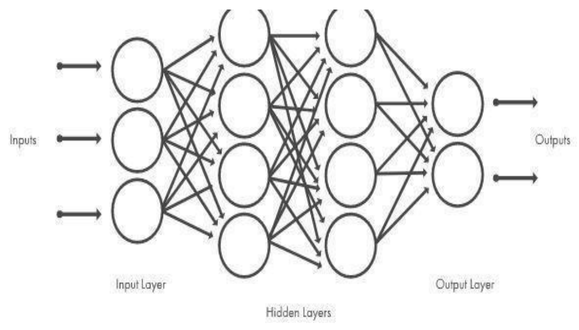


Fig 2.1 Deep Neural network

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 is 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.

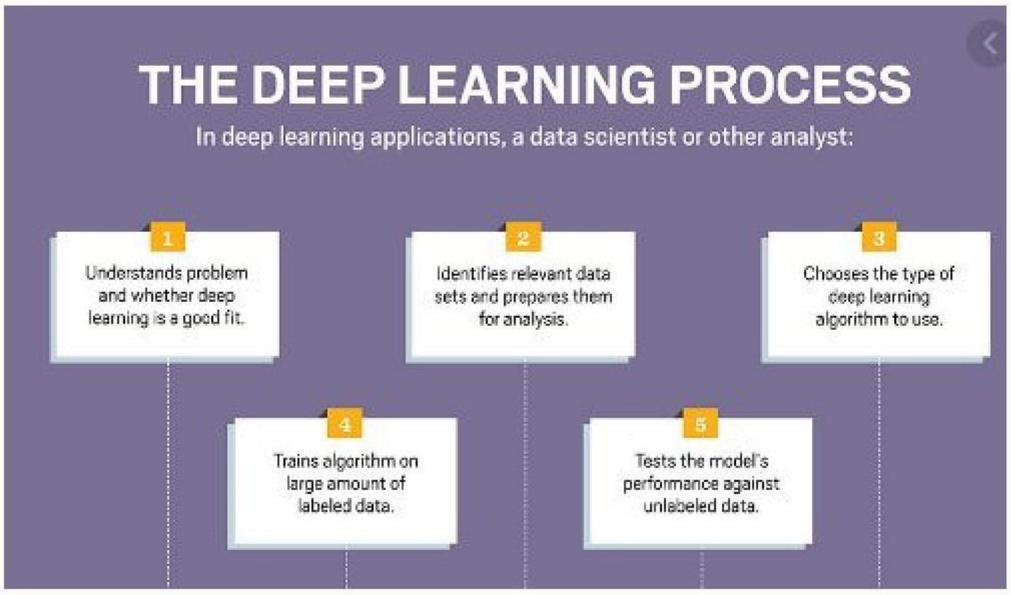

THE DEEP LEARNING PROCESS
In deep learning applications, a data scientist or other analyst:
1 Understands problem and whether deep learning is a good fit.
2 Identifies relevant data sets and prepares them for analysis.
3 Chooses the type of deep learning algorithm to use.
4 Trains algorithm on large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

Fig 2.2 The deep learning process

Relation between Data Mining, Machine Learning and Deep Learning:

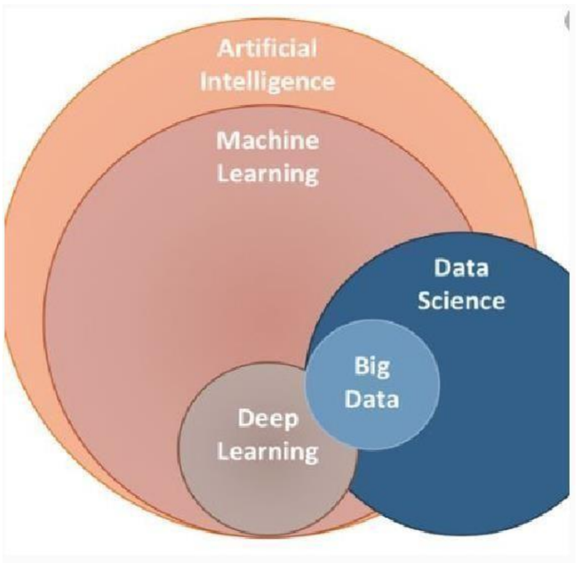


Fig 2.3 Relation between DM, ML, DL

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.

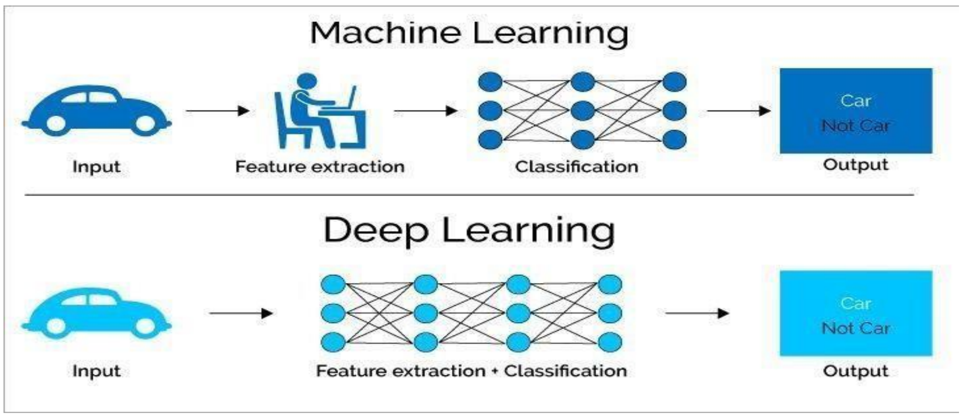


Fig 2.3 process in machine learning and deep learning

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

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.

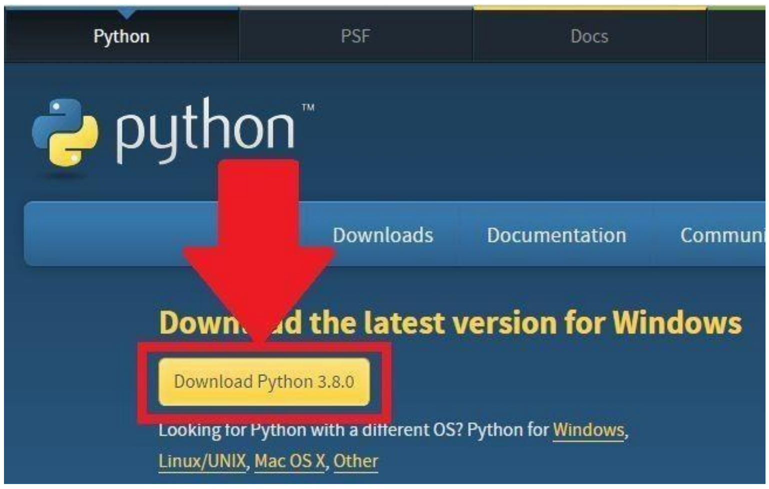
Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for Use as a scripting or glue language to 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.

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/

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.

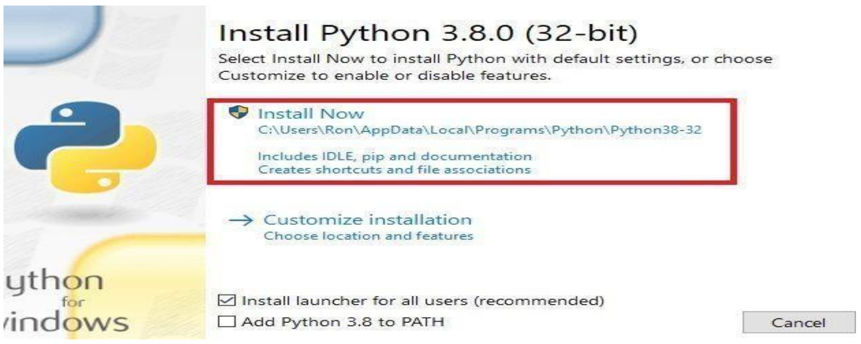


Download the latest version for Windows

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.

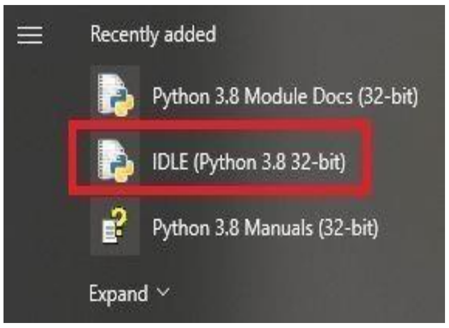
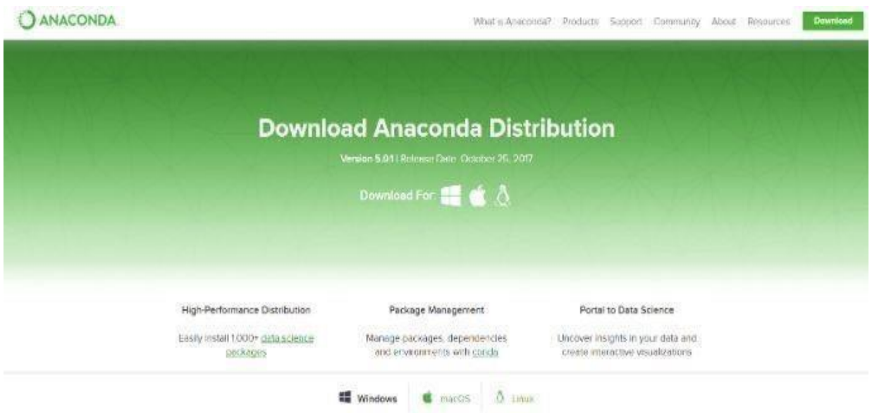


Fig 3.2.1.2 IDLE

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:

  i. Go to the following link: Anaconda.com/downloads

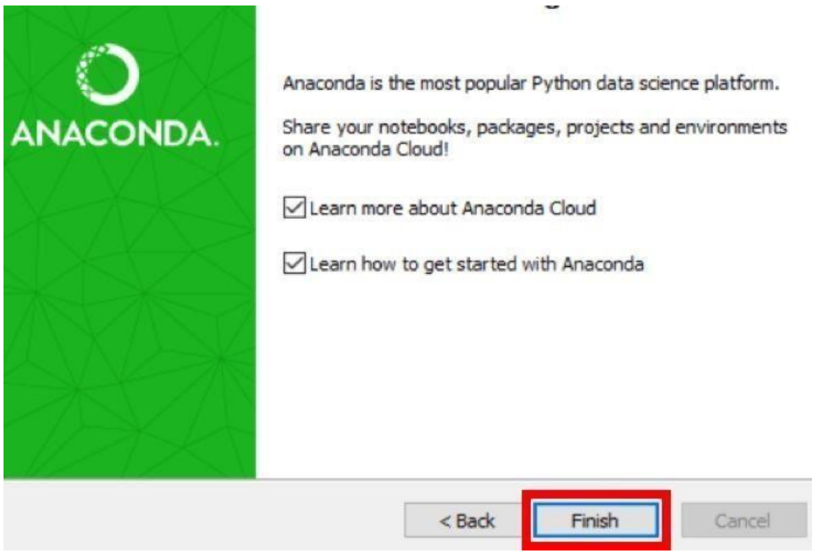

ANACONDA
What is Anaconda? Products Support Community About Resources Download
Download Anaconda Distribution
Version 5.0.1 | Release Date: October 25, 2017
Download For:
High-Performance Distribution
Easily install 1,000+ data science packages
Package Management
Manage packages, dependencies and environments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

ii. Download python 3.4 version for (32-bitgraphic installer/64 -bit graphic installer)

iii. Select path(i.e. add anaconda to path & register anaconda as default python 3.4)

iv. Click finish

v. Open jupyter notebook



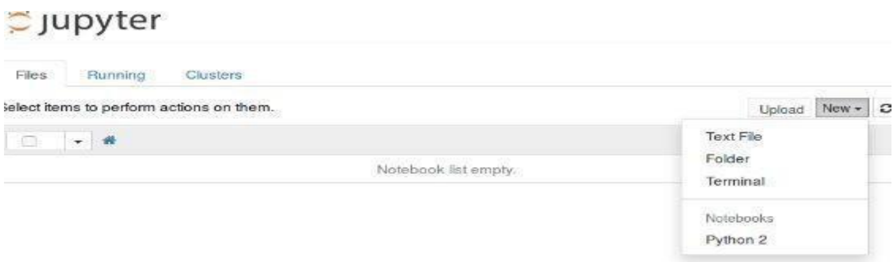
Anaconda is the most popular Python data science platform.

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



Select items to perform actions on them.

Notebook list empty.

Fig 3.2.2.2 jupyter notebook

Features:

i. **Readable:** Python is a very readable language.

ii. **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

iii. **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.

iv. **Open Source:** Python is a open source programming language.

v. **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.

vi. **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.

vii. **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.

viii. **Advanced features:** Supports generators and list comprehensions. We will cover these features later.

ix. **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.

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

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 = 2.5 x 10^2= 250).

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".

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 inpython 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.

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

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 ofthe keys we declare in a dictionary are always unique, we can use them as indexes to access the elements in a dictionary.

Functions:

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.

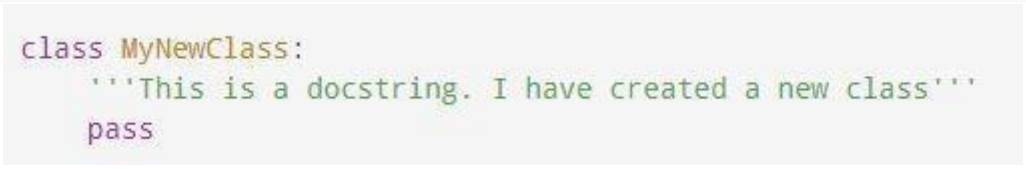
Calling a Function:

- 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

OOPs Concepts:

3.6.1 Class:

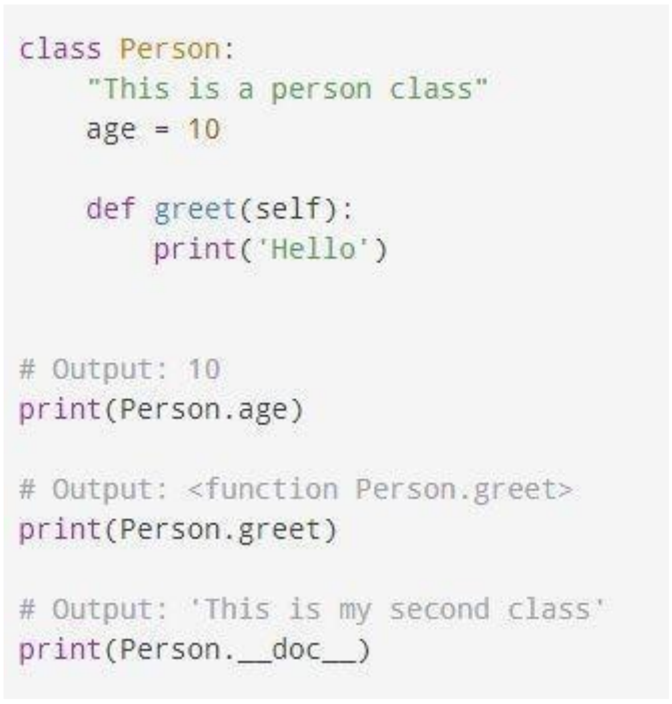
- Python is an object oriented programming language. Unlike procedure oriented programming, where the main emphasis is on functions, object oriented programming stresses on objects..
- An object is simply a collection of data (variables) and methods (functions) that act onthose data. Similarly, a class 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 called **instantiation**.
- 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 MyNewClass:
    '''This is a docstring. I have created a new class'''
    pass
```

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. OBJECT DETECTION:

INTRODUCTION:

1. A few years ago, the creation of the software and hardware image processing systems was mainly limited to the development of the user interface , which most of the programmers of each firm were engaged in . The situation has been significantly changed with the advent of the Windows operating system when the majority of the developers switched to solving the problems of image processing itself . How ever , this has not yet led to the cardinal progress in solving typical tasks of recognizing faces , car numbers , road signs , analyzing remote and medical images , etc .
2. Each of these "eternal" problems is solved by trial and error by the efforts of numerous groups of the engineers and scientists . As modern technical solutions are turn out to be excessively expensive , the task of automating the creationof the software tools for solving intellectual problems is formulated and intensively solved abroad . In the field of image processing , the required toolkit should be supporting the analysis and recognition of images of previously unknown content and ensure the effective development tof applications by ordinary programmers . Just as the Windows tool kit supports the creation of interfaces for solving various applied problems.

3. Object recognition is to describe a collection of related computer vision tasks that involve activities like identifying objects in digital photographs . Image classification involves activities such as predicting the class of one object in an image . Object localization is refers to identifying the location of one or more objects in an image and drawing an a bounding box around their extent . Object detection does the work of combines these two tasks and localizes and classifies one or more objects in an image . When a user or practitioner refers to the term “object recognition“ , they often mean “object detection“ .

- PROBLEM STATEMENT :

Project is about checking out the images and able to detect the object ,that is Aeroplane detection .That means trying to detect whether my images has aeroplane or not with the accuracy.

ROAD MAP:

4.1 GATHERING IMAGES AND LABELS:

Preprocessing of the data actually involves the following steps:

4.1.1 GETTING THE DATASET:

- The data set is downloaded from the chrome that contain's images and the csv files(annotations)
- The data set contains the aeroplane as an images and its annotations

4.1.2 Display of the images and csv files(annotations)

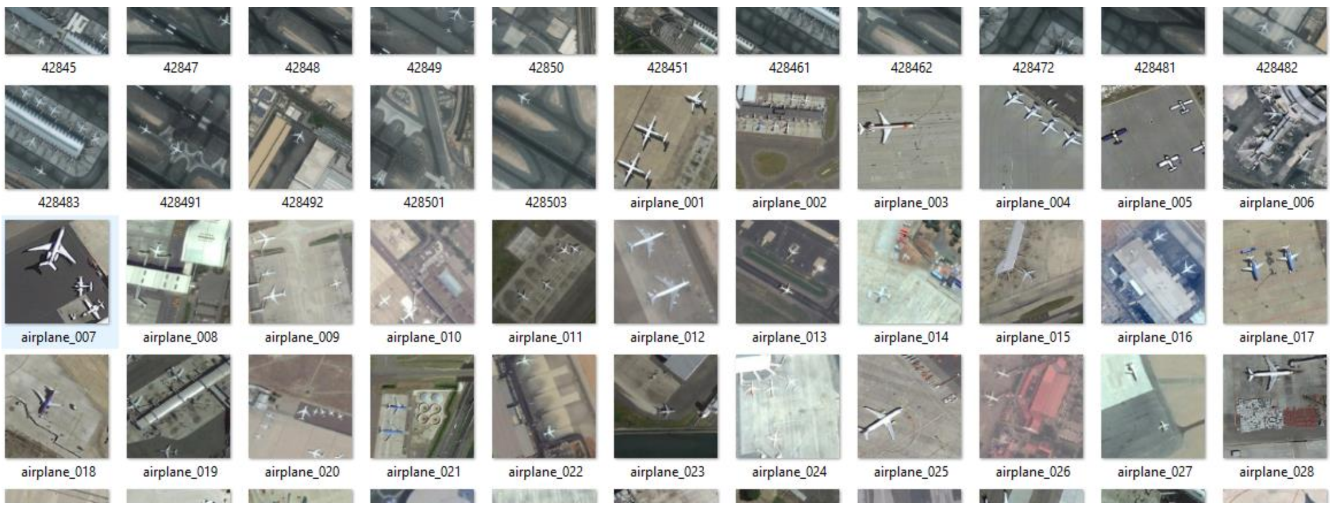
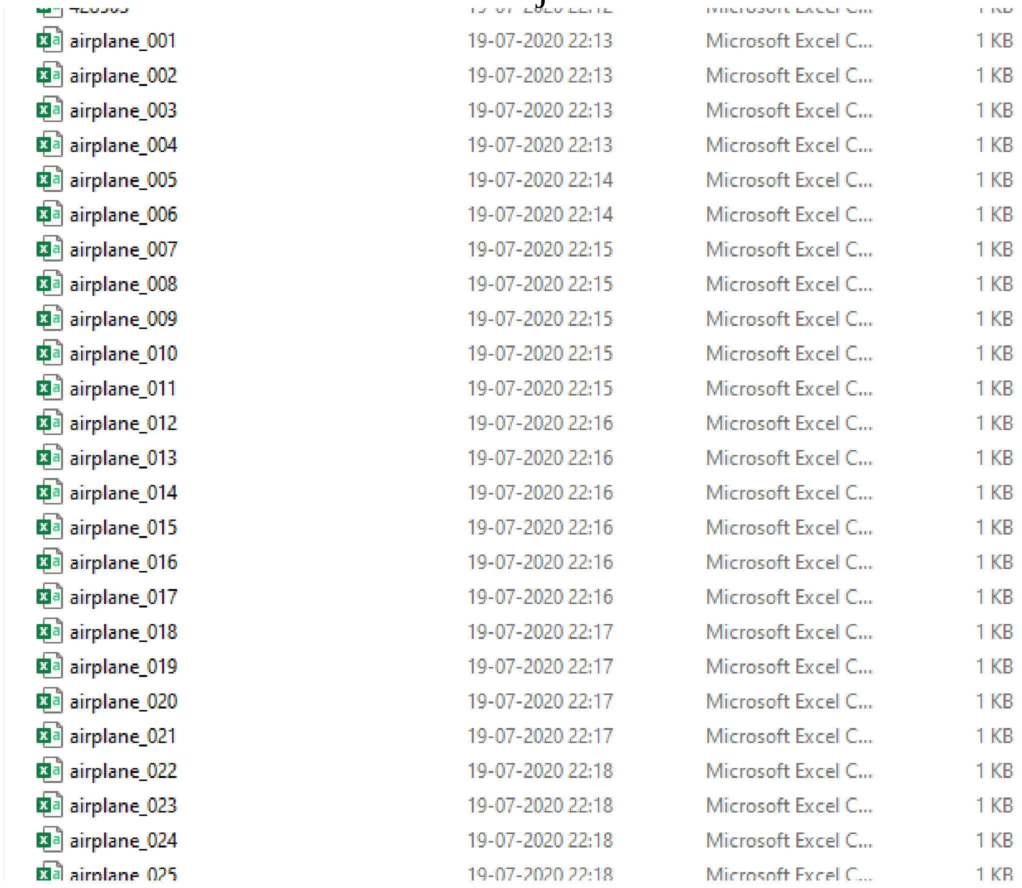


Figure 4.1.2.1 example of aeroplane images



airplane_001	19-07-2020 22:13	Microsoft Excel C...	1 KB
airplane_002	19-07-2020 22:13	Microsoft Excel C...	1 KB
airplane_003	19-07-2020 22:13	Microsoft Excel C...	1 KB
airplane_004	19-07-2020 22:13	Microsoft Excel C...	1 KB
airplane_005	19-07-2020 22:14	Microsoft Excel C...	1 KB
airplane_006	19-07-2020 22:14	Microsoft Excel C...	1 KB
airplane_007	19-07-2020 22:15	Microsoft Excel C...	1 KB
airplane_008	19-07-2020 22:15	Microsoft Excel C...	1 KB
airplane_009	19-07-2020 22:15	Microsoft Excel C...	1 KB
airplane_010	19-07-2020 22:15	Microsoft Excel C...	1 KB
airplane_011	19-07-2020 22:15	Microsoft Excel C...	1 KB
airplane_012	19-07-2020 22:16	Microsoft Excel C...	1 KB
airplane_013	19-07-2020 22:16	Microsoft Excel C...	1 KB
airplane_014	19-07-2020 22:16	Microsoft Excel C...	1 KB
airplane_015	19-07-2020 22:16	Microsoft Excel C...	1 KB
airplane_016	19-07-2020 22:16	Microsoft Excel C...	1 KB
airplane_017	19-07-2020 22:16	Microsoft Excel C...	1 KB
airplane_018	19-07-2020 22:17	Microsoft Excel C...	1 KB
airplane_019	19-07-2020 22:17	Microsoft Excel C...	1 KB
airplane_020	19-07-2020 22:17	Microsoft Excel C...	1 KB
airplane_021	19-07-2020 22:17	Microsoft Excel C...	1 KB
airplane_022	19-07-2020 22:18	Microsoft Excel C...	1 KB
airplane_023	19-07-2020 22:18	Microsoft Excel C...	1 KB
airplane_024	19-07-2020 22:18	Microsoft Excel C...	1 KB
airplane_025	19-07-2020 22:18	Microsoft Excel C...	1 KB

Figure 4.1.2.2 display of annotations files

- As we need to label the images to get the values as boundary box
- As I have the boundary boxes of my aeroplane images , so I have not done labelling for my images
- The boundary boxes are present in the csv files

4.2 Setting the environment:

4.2.1 Set up The Google Colab Notebook:

1. Create a new Notebook.

2. From the top left menu: Go to **Runtime** > **Change runtime type** > select **GPU** from **hardware accelerator.** Some pretrained models support TPU. The pretrained model we are choosing in this project only supports GPU.

3. (Highly Recommended) Mount Google Drive to the Colab notebook:

   - When training starts, checkpoints, logs and many other important files will be created. When the kernel disconnects, these files, along with everything else, will be deleted if they don't get saved on your Google Drive or somewhere else.

   - The kernel disconnects shortly after your computer sleeps or after using the Colab **GPU** for 12 hours. Training will need to be restarted from zero if the trained model did not get saved.

Mounting the google drive :

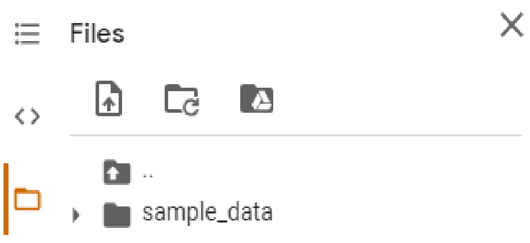
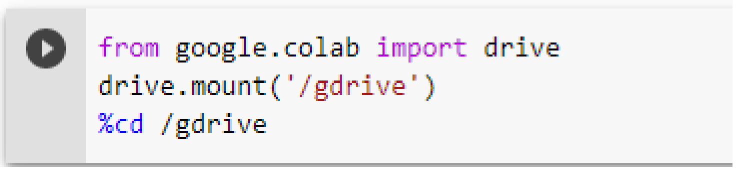


Figure 4.2.1 pic of mount of drive

The command for mount the drive is :

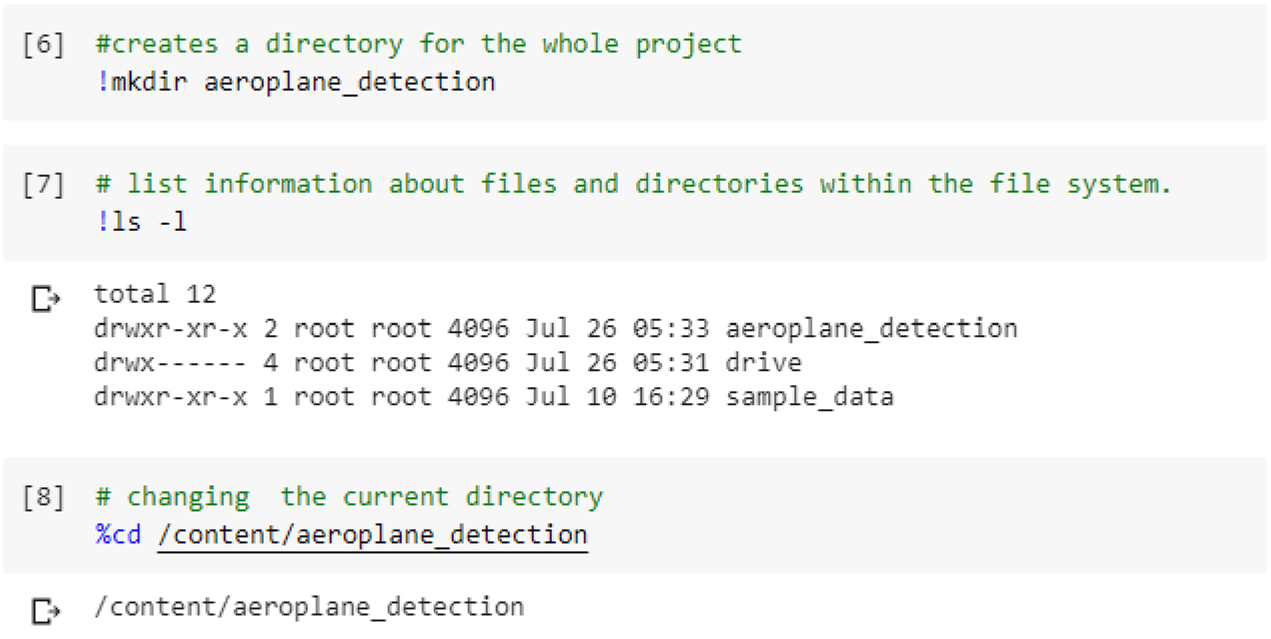


```
from google.colab import drive
drive.mount('/gdrive')
%cd /gdrive
```

Figure 4.2.2 mounting the drive

2.Upload your images and labels:

1. Creating the directory:



```
[6]  #creates a directory for the whole project
     !mkdir aeroplane_detection
```

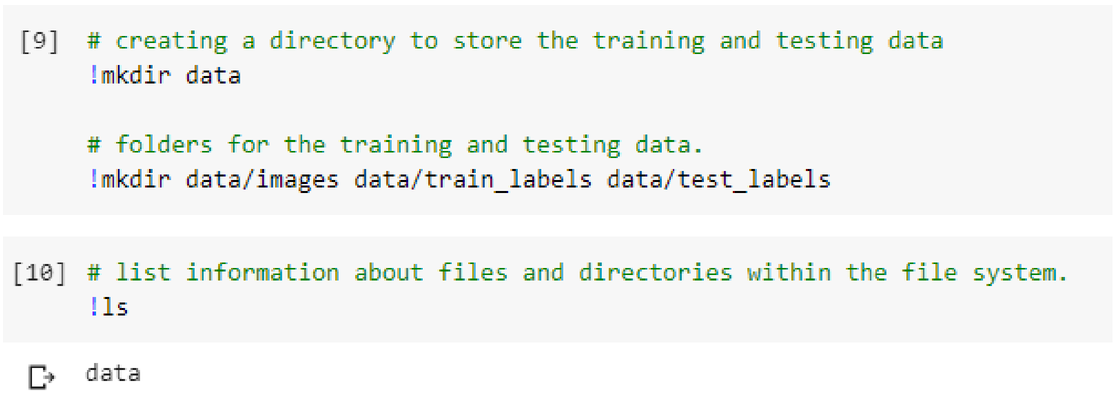
```
[7]  # list information about files and directories within the file system.
     !ls -l
```

```
total 12
drwxr-xr-x 2 root root 4096 Jul 26 05:33 aeroplane_detection
drwx------ 4 root root 4096 Jul 26 05:31 drive
drwxr-xr-x 1 root root 4096 Jul 10 16:29 sample_data
```

```
[8]  # changing  the current directory
     %cd /content/aeroplane_detection
```

```
/content/aeroplane_detection
```

2. Creating the data folder under it and creating the folder under it as images,train_labels and the test_labels

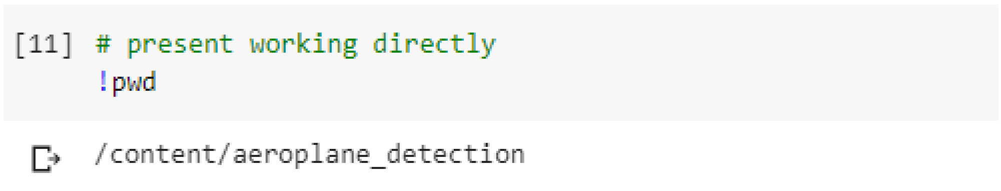


```
[9]  # creating a directory to store the training and testing data
     !mkdir data

     # folders for the training and testing data.
     !mkdir data/images data/train_labels data/test_labels
```

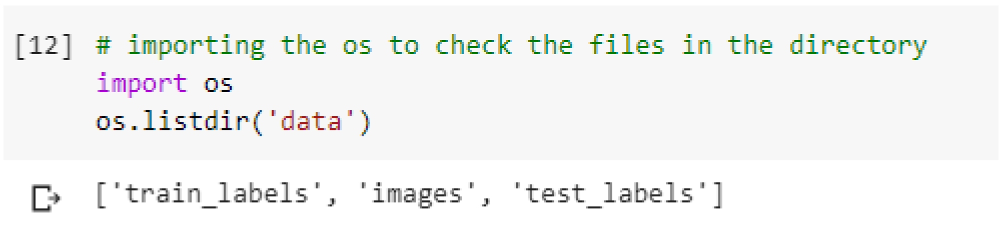
```
[10] # list information about files and directories within the file system.
     !ls
```

```
data
```

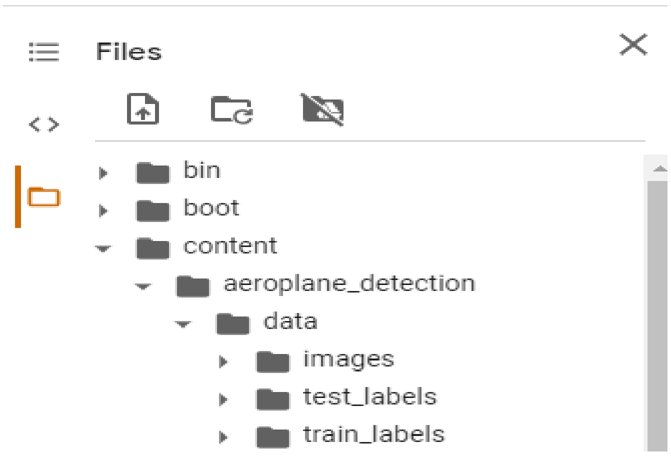
3. Checking the present directory and importing the OS:



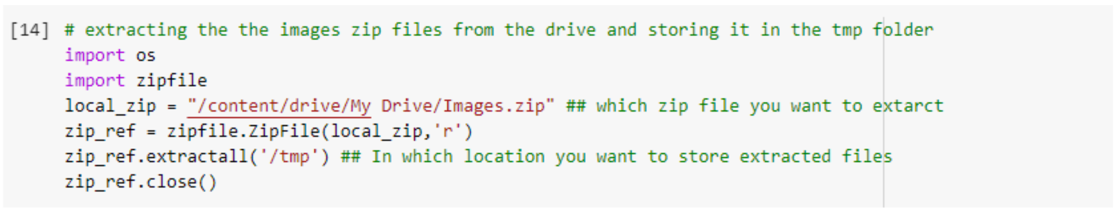
```
[11] # present working directly
     !pwd

     /content/aeroplane_detection
```



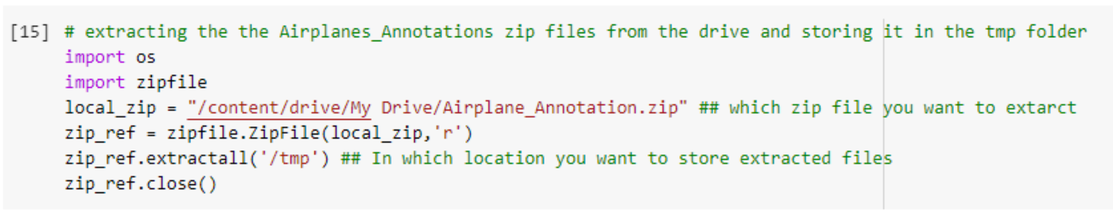
```
[12] # importing the os to check the files in the directory
     import os
     os.listdir('data')

     ['train_labels', 'images', 'test_labels']
```



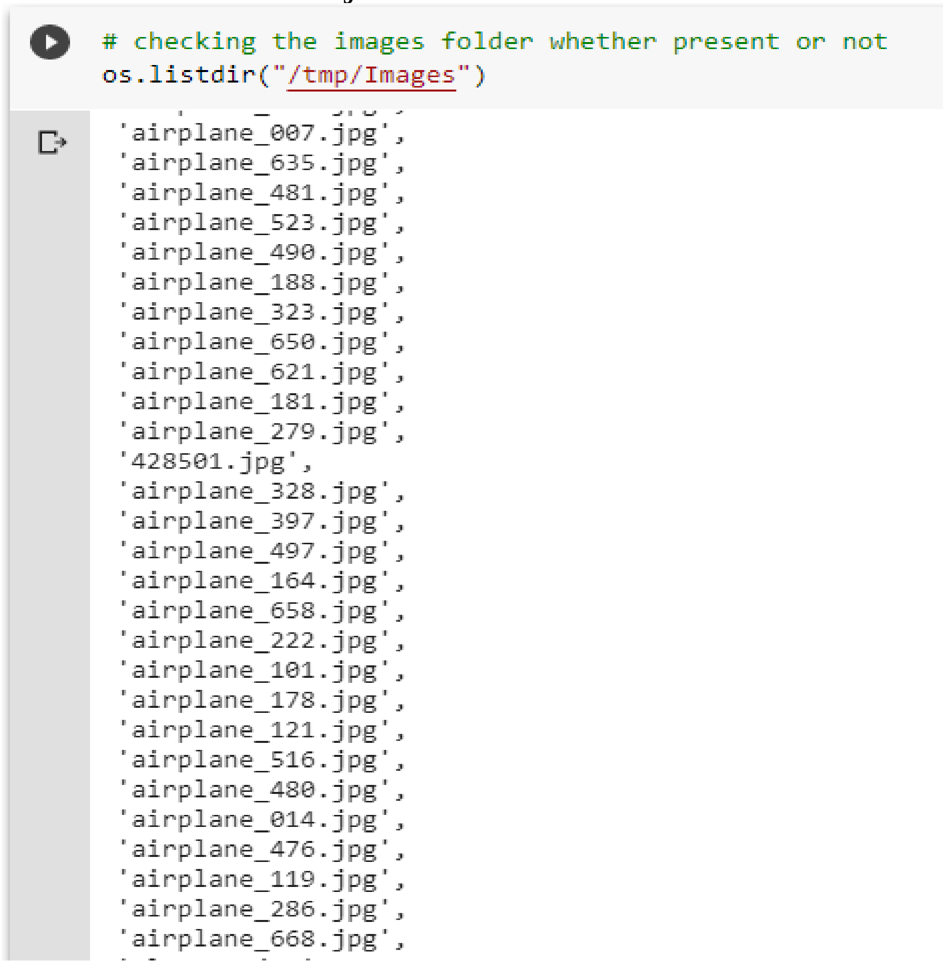
4. I have stored my images and annotations zip files in my drive . So I have to unzip , through the unzip command into the tmp folder to access my files . Checking the images and the annotations present in the tmp folder.



```
[14] # extracting the the images zip files from the drive and storing it in the tmp folder
     import os
     import zipfile
     local_zip = "/content/drive/My Drive/Images.zip" ## which zip file you want to extarct
     zip_ref = zipfile.ZipFile(local_zip,'r')
     zip_ref.extractall('/tmp') ## In which location you want to store extracted files
     zip_ref.close()
```

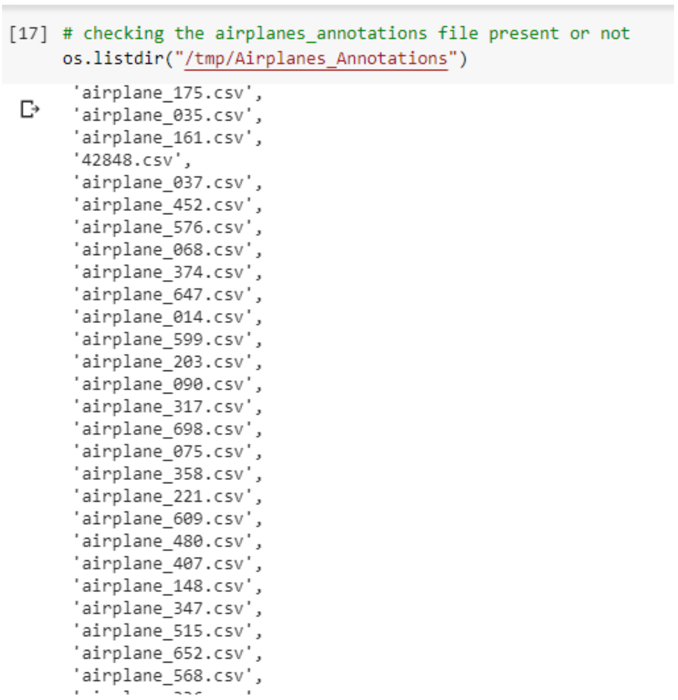


```
[15] # extracting the the Airplanes_Annotations zip files from the drive and storing it in the tmp folder
     import os
     import zipfile
     local_zip = "/content/drive/My Drive/Airplane_Annotation.zip" ## which zip file you want to extarct
     zip_ref = zipfile.ZipFile(local_zip,'r')
     zip_ref.extractall('/tmp') ## In which location you want to store extracted files
     zip_ref.close()
```



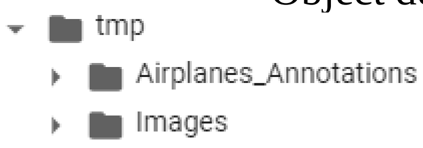
```
# checking the images folder whether present or not
os.listdir("/tmp/Images")
```

```
 'airplane_007.jpg',
 'airplane_635.jpg',
 'airplane_481.jpg',
 'airplane_523.jpg',
 'airplane_490.jpg',
 'airplane_188.jpg',
 'airplane_323.jpg',
 'airplane_650.jpg',
 'airplane_621.jpg',
 'airplane_181.jpg',
 'airplane_279.jpg',
 '428501.jpg',
 'airplane_328.jpg',
 'airplane_397.jpg',
 'airplane_497.jpg',
 'airplane_164.jpg',
 'airplane_658.jpg',
 'airplane_222.jpg',
 'airplane_101.jpg',
 'airplane_178.jpg',
 'airplane_121.jpg',
 'airplane_516.jpg',
 'airplane_480.jpg',
 'airplane_014.jpg',
 'airplane_476.jpg',
 'airplane_119.jpg',
 'airplane_286.jpg',
 'airplane_668.jpg',
```

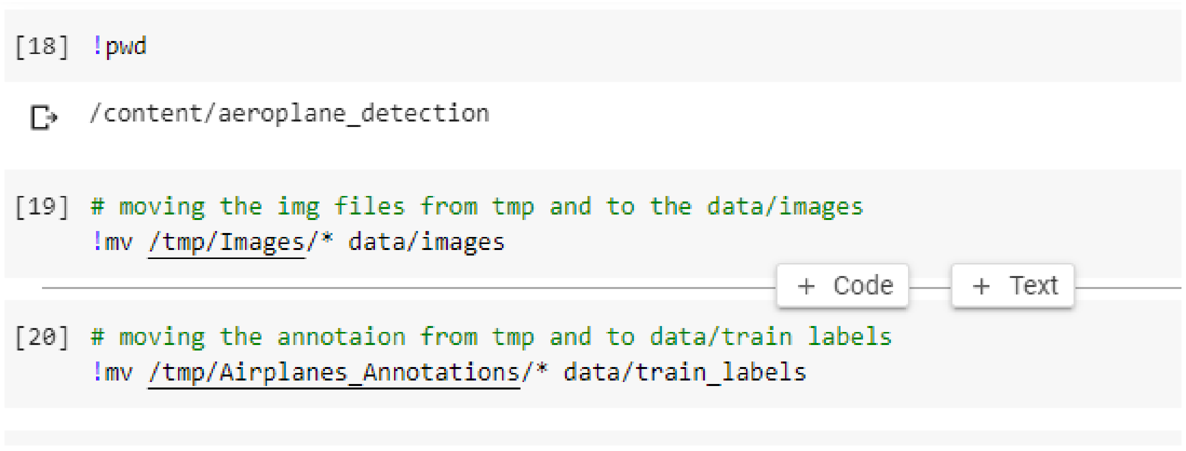


```
[17] # checking the airplanes_annotations file present or not
     os.listdir("/tmp/Airplanes_Annotations")
```

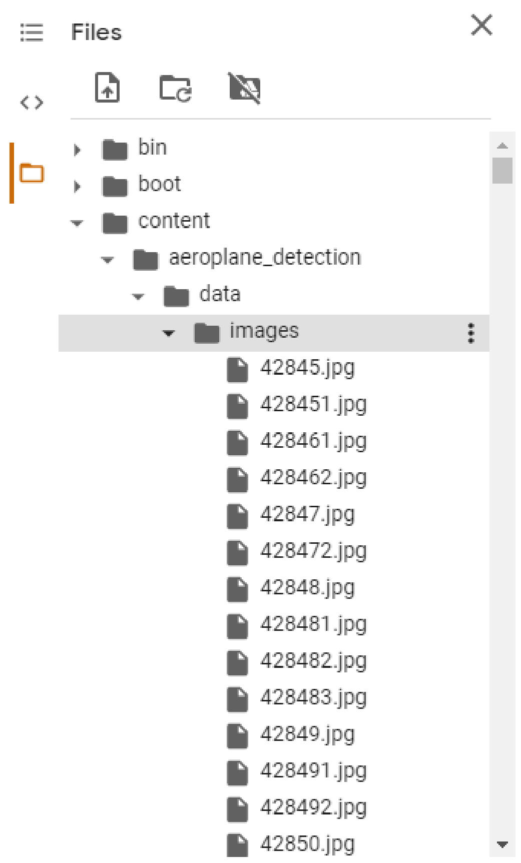
```
 'airplane_175.csv',
 'airplane_035.csv',
 'airplane_161.csv',
 '42848.csv',
 'airplane_037.csv',
 'airplane_452.csv',
 'airplane_576.csv',
 'airplane_068.csv',
 'airplane_374.csv',
 'airplane_647.csv',
 'airplane_014.csv',
 'airplane_599.csv',
 'airplane_203.csv',
 'airplane_090.csv',
 'airplane_317.csv',
 'airplane_698.csv',
 'airplane_075.csv',
 'airplane_358.csv',
 'airplane_221.csv',
 'airplane_609.csv',
 'airplane_480.csv',
 'airplane_407.csv',
 'airplane_148.csv',
 'airplane_347.csv',
 'airplane_515.csv',
 'airplane_652.csv',
 'airplane_568.csv',
```



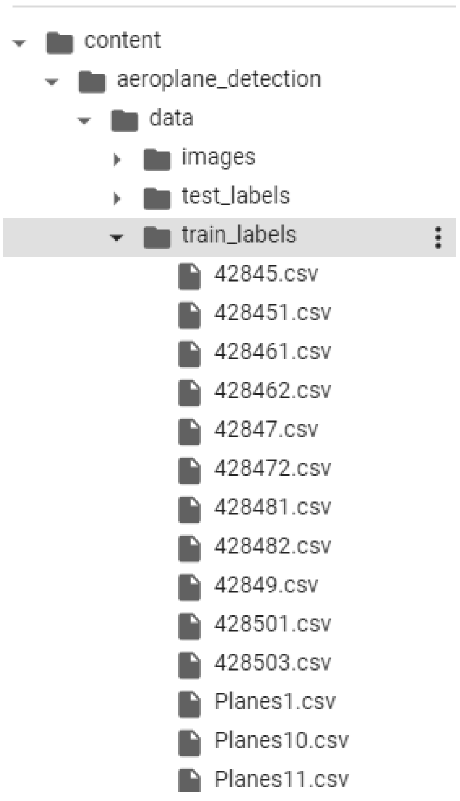
5. Moving my Images and the annotations folders from the tmp folder to the newly created the folders. There are the images and train_labels.



```
[18] !pwd

/content/aeroplane_detection

[19] # moving the img files from tmp and to the data/images
     !mv /tmp/Images/* data/images

[20] # moving the annotaion from tmp and to data/train labels
     !mv /tmp/Airplanes_Annotations/* data/train_labels
```

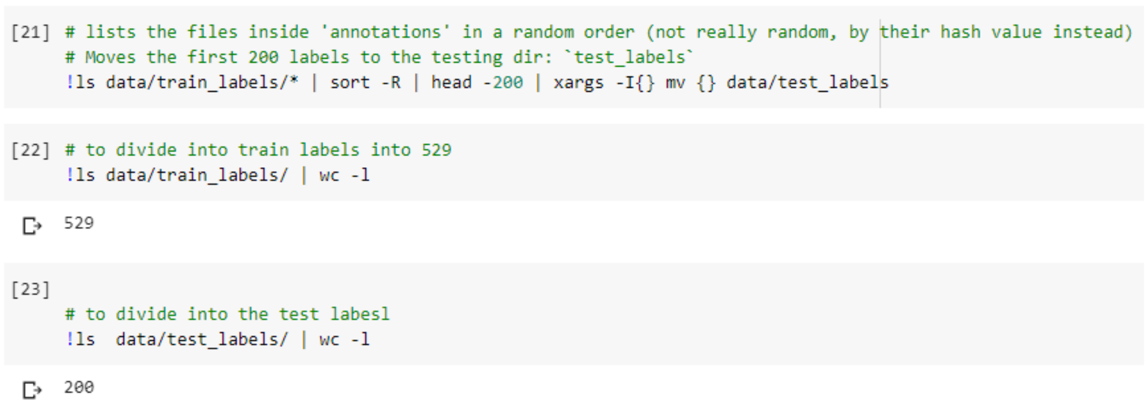


bin
boot
content
aeroplane_detection
data
images
42845.jpg
428451.jpg
428461.jpg
428462.jpg
42847.jpg
428472.jpg
42848.jpg
428481.jpg
428482.jpg
428483.jpg
42849.jpg
428491.jpg
428492.jpg
42850.jpg


content
aeroplane_detection
data
images
test_labels
train_labels
42845.csv
428451.csv
428461.csv
428462.csv
42847.csv
428472.csv
428481.csv
428482.csv
42849.csv
428501.csv
428503.csv
Planes1.csv
Planes10.csv
Planes11.csv

3.Splitting the images into training & testing:

1. we need to split the files into the train_labels and test_labels .Moving the files from the files from the train_labels to test labels .As my Images and Annotations files consists of 729 images in image folder and the its 729 annotaions in train_labels.

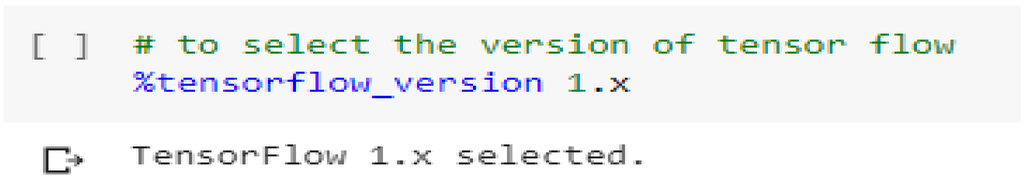


```
[21] # lists the files inside 'annotations' in a random order (not really random, by their hash value instead)
     # Moves the first 200 labels to the testing dir: `test_labels`
     !ls data/train_labels/* | sort -R | head -200 | xargs -I{} mv {} data/test_labels

[22] # to divide into train labels into 529
     !ls data/train_labels/ | wc -l

     529

[23]
     # to divide into the test labesl
     !ls  data/test_labels/ | wc -l

     200
```

Splitted the train_labels into 529 and test_labels into 200 files.

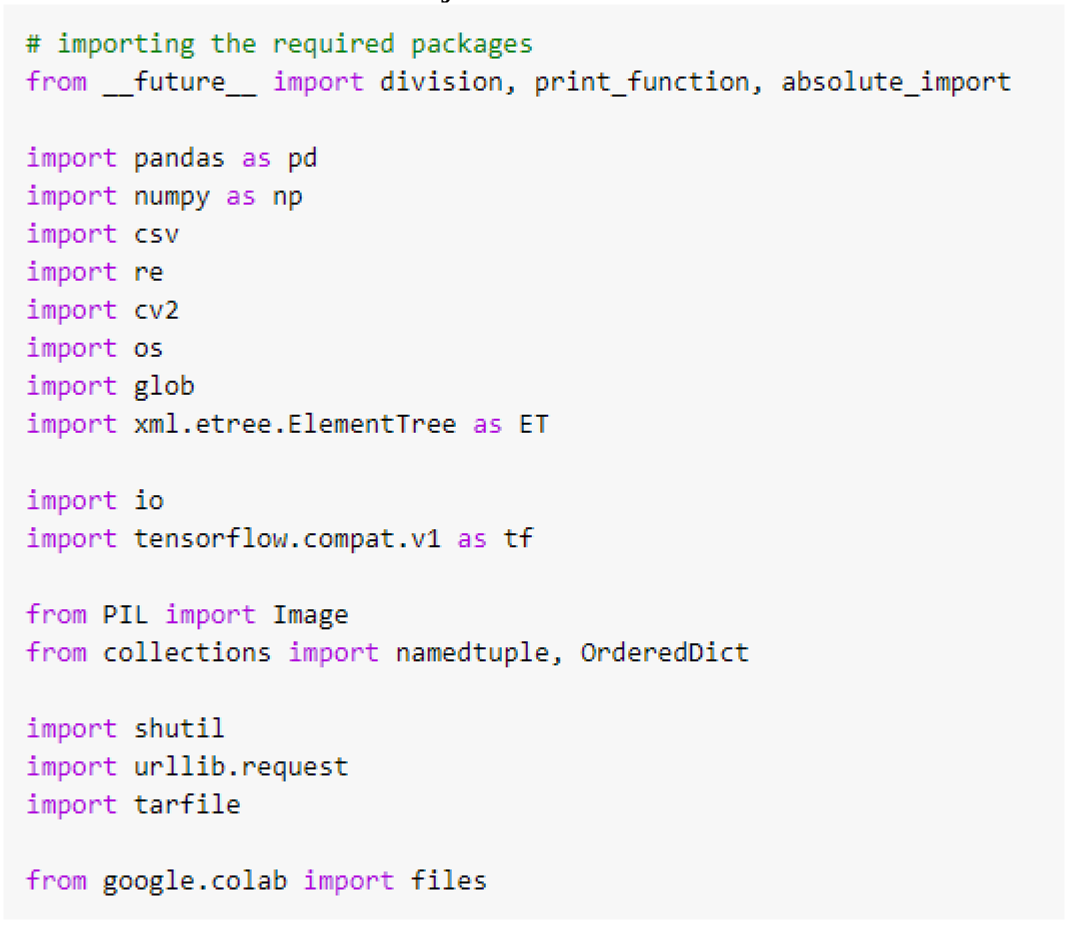
4.3 **Importing and Installing Required Packages:**

- Selecting the tensor flow version



```
[ ]  # to select the version of tensor flow
     %tensorflow_version 1.x

     TensorFlow 1.x selected.
```

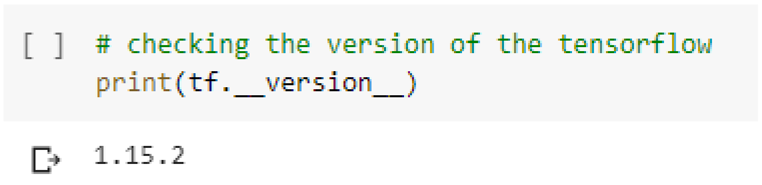
- Here we are importing all the packages that are required for the project



```
# importing the required packages
from __future__ import division, print_function, absolute_import

import pandas as pd
import numpy as np
import csv
import re
import cv2
import os
import glob
import xml.etree.ElementTree as ET

import io
import tensorflow.compat.v1 as tf

from PIL import Image
from collections import namedtuple, OrderedDict

import shutil
import urllib.request
import tarfile

from google.colab import files
```

Fig 4.1.1 packages

- **Checking the version of the tensor flow**:



```
[ ]  # checking the version of the tensorflow
     print(tf.__version__)
```

```
1.15.2
```

We have to import the libraries as per the requirement of the algorithm.

Technology Used:

- **Google colab**: Google Colab is a free cloud service and now it supports free GPU! You can , improve your Python programming language coding skills. develop deep learning applications using popular libraries such as Keras, TensorFlow, PyTorch, and OpenCV.

Project Requirements:

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.

- **CSV:** The so-called CSV (Comma Separated Values) format is the most common import and export format for spreadsheets and databases**.**
  The **csv** module implements classes to read and write tabular data in **CSV** format.
- **Re :** A **regular expression** is a special sequence of characters that helps you match or find other strings or sets of strings, using a specialized syntax held in a pattern.

- **CV2:** **OpenCV-Python** is a library of **Python** bindings designed to solve computer vision problems. **cv2**. imread() method loads an image from the specified file.

- **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.
- **OS :** The **OS module** in **Python** provides a way of using **operating system** dependent functionality. The functions that the **OS module** provides allows you to interface with the underlying **operating system** that **Python** is running on – be that Windows, Mac or Linux.
- **GLOB : Glob** is a general term used to **define** techniques to match specified patterns according to rules related to Unix shell. ... In **Python**, the **glob** module is used to retrieve files/pathnames matching a specified pattern. The pattern rules of **glob** follow standard Unix path expansion rules.
- **Xml**.etree : **xml**.**etree**.**ElementTree**. **XML** (text, parser=None) Parses an **XML** section from a string constant. This function can be used to embed “**XML** literals” in **Python** code. text is a string containing **XML** data.

- **IO : io** — Core tools for working with streams. The **io** module provides the **Python** interfaces to stream handling.

- **tensorflow.compat.v1:** autograph module: Conversion of plain **Python** into **TensorFlow** graph code. Generalization of **tf.compat.v1**.scatter_update to axis different than 0. (deprecated) . Computes the **mean** of elements across dimensions of a tensor.

- **PIL import Image:** how to load and preprocess an **image** . Reads the images

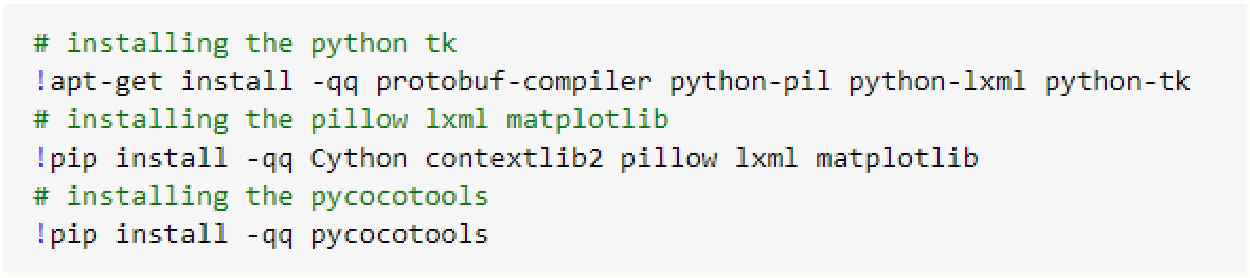
- **Collections:** **Collections in Python** are containers that are used to store **collections** of data, for example, list, dict, set, tuple.

- **Shutil:** **Python shutil**. **Python shutil** module enables us to operate with file objects easily and without diving into file objects a lot. It takes care of low-level semantics like creating file objects, closing the files once they are copied and allows us to focus on the business logic of our program.

- **urllib**. request : **urllib**. request is a **Python** module for fetching URLs (Uniform Resource Locators). ... This is capable of fetching URLs using a variety of different protocols. It also offers a slightly more complex interface for handling common situations - like basic authentication, cookies, proxies

- **tarfile :** The **tarfile** module makes it possible to read and write tar archives, including those using gzip or bz2 compression. Use the zipfile module to read or write . zip files, or the higher-level functions in shutil.reads and writes gzip and bz2 compressed archives if the respective modules are available.

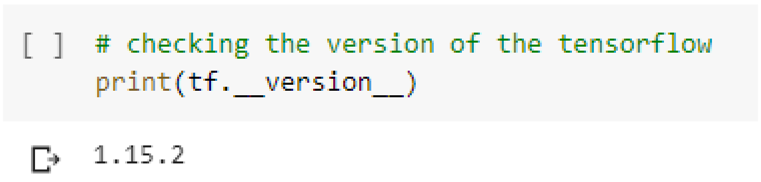
- Installing the required packages :



```
# installing the python tk
!apt-get install -qq protobuf-compiler python-pil python-lxml python-tk
# installing the pillow lxml matplotlib
!pip install -qq Cython contextlib2 pillow lxml matplotlib
# installing the pycocotools
!pip install -qq pycocotools
```

4.3.1 Versions of the packages:

The versions of the packages are found by following command

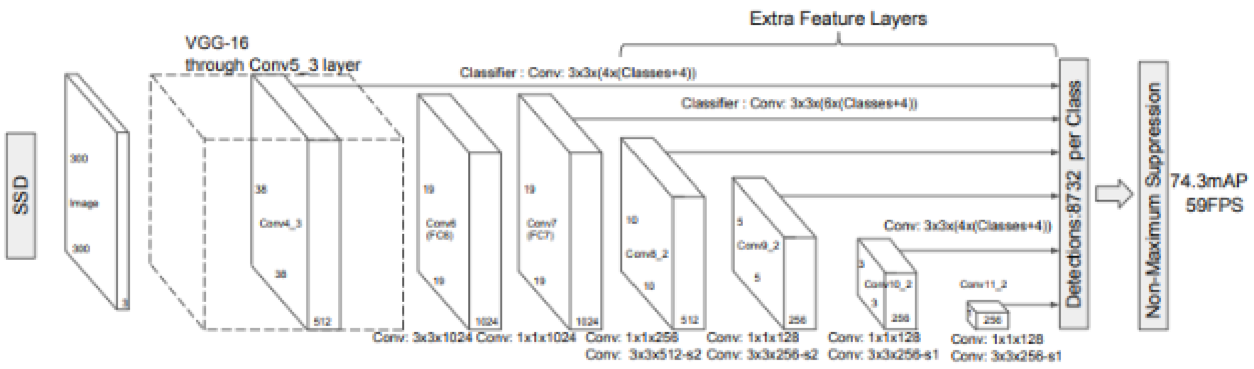


```
[ ] # checking the version of the tensorflow
    print(tf.__version__)

1.15.2
```

Fig 4.1.2 version

4.3.2 Model used :

SSD: SINGLE SHOT DETECTOR:



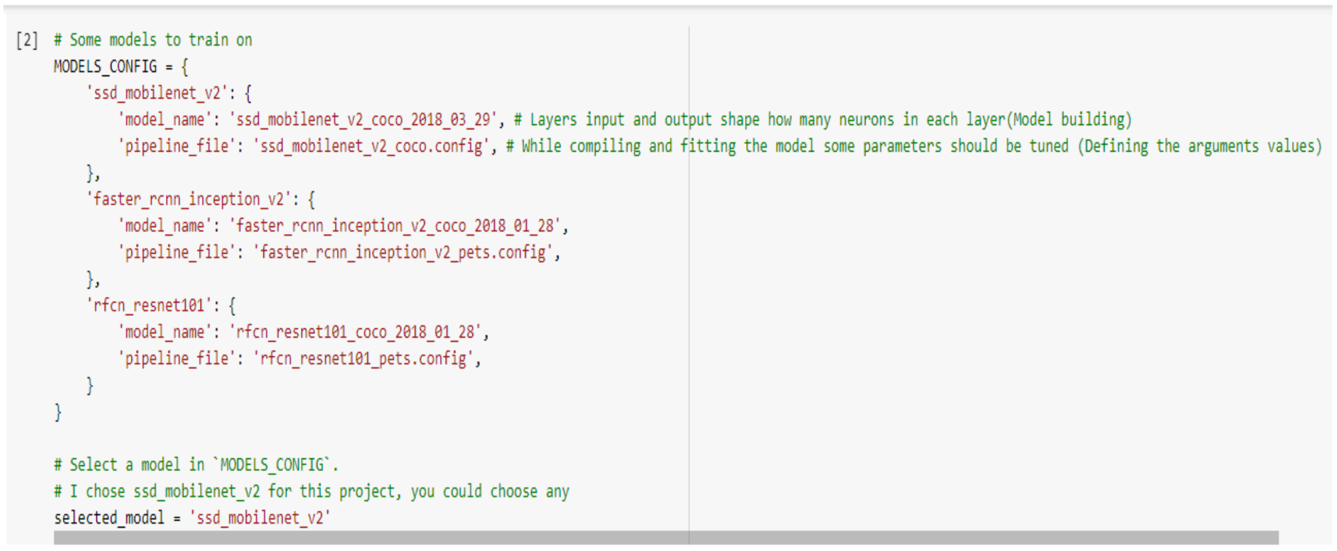
- Single Shot Detector (SSD) is a method for detecting objects in images using a single deep neural network. The SSD approach discretises the output space of bounding boxes into a set of default boxes over different aspect ratios. After discretising, the method scales per feature map location. The Single Shot Detector network combines predictions from multiple feature maps with different resolutions to naturally handle objects of various sizes.

- Adavantages of ssd:

  i. SSD completely eliminates proposal generation and subsequent pixel or feature resampling stages and encapsulates all computation in a single network.

ii. Easy to train and straightforward to integrate into systems that require a detection component.

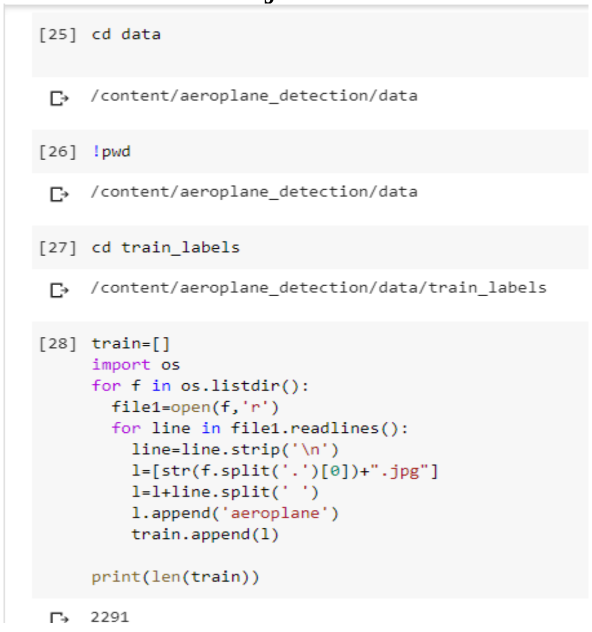
iii. SSD has competitive accuracy to methods that utilize an additional object proposal step, and it is much faster while providing a unified framework for both training and inference.



```
[2] # Some models to train on
    MODELS_CONFIG = {
        'ssd_mobilenet_v2': {
            'model_name': 'ssd_mobilenet_v2_coco_2018_03_29', # Layers input and output shape how many neurons in each layer(Model building)
            'pipeline_file': 'ssd_mobilenet_v2_coco.config', # While compiling and fitting the model some parameters should be tuned (Defining the arguments values)
        },
        'faster_rcnn_inception_v2': {
            'model_name': 'faster_rcnn_inception_v2_coco_2018_01_28',
            'pipeline_file': 'faster_rcnn_inception_v2_pets.config',
        },
        'rfcn_resnet101': {
            'model_name': 'rfcn_resnet101_coco_2018_01_28',
            'pipeline_file': 'rfcn_resnet101_pets.config',
        }
    }

    # Select a model in `MODELS_CONFIG`.
    # I chose ssd_mobilenet_v2 for this project, you could choose any
    selected_model = 'ssd_mobilenet_v2'
```

4.4 Preprocessing Images and Labels:

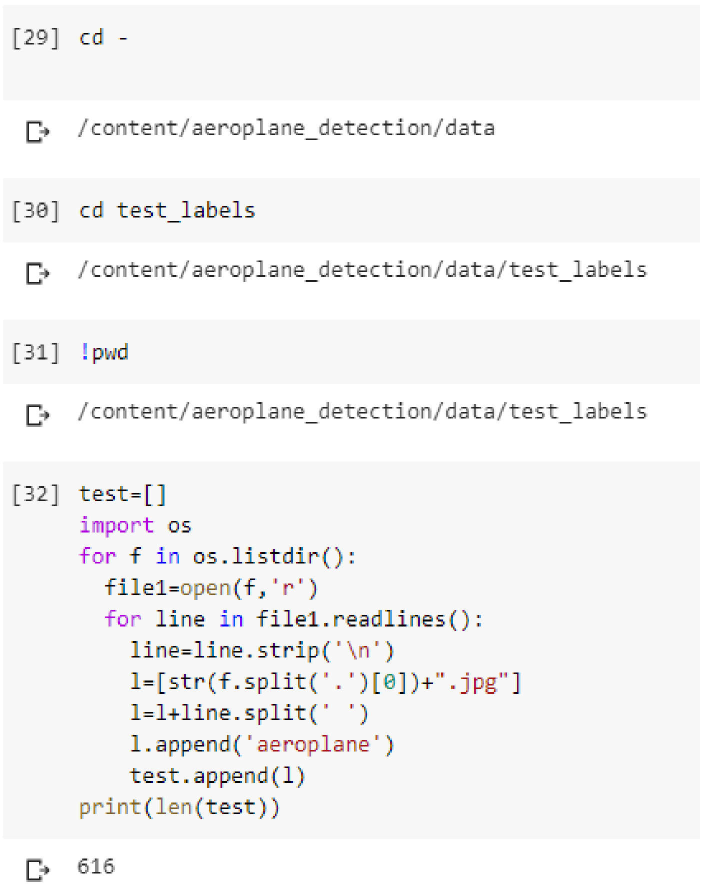
- As we need to convert all files in to DataFrame . So i am converting into the list as ‘train[]’ and ‘test[]’ .And here I am running a for loop to store all files into list and to add the files name as .jpg for the column through the ‘spilt()’. Appending the aeroplane column , i don’t have my class column in my dataframe .



```
[25] cd data

/content/aeroplane_detection/data

[26] !pwd

/content/aeroplane_detection/data

[27] cd train_labels

/content/aeroplane_detection/data/train_labels

[28] train=[]
     import os
     for f in os.listdir():
       file1=open(f,'r')
       for line in file1.readlines():
         line=line.strip('\n')
         l=[str(f.split('.')[0])+".jpg"]
         l=l+line.split(' ')
         l.append('aeroplane')
         train.append(l)

     print(len(train))

2291
```

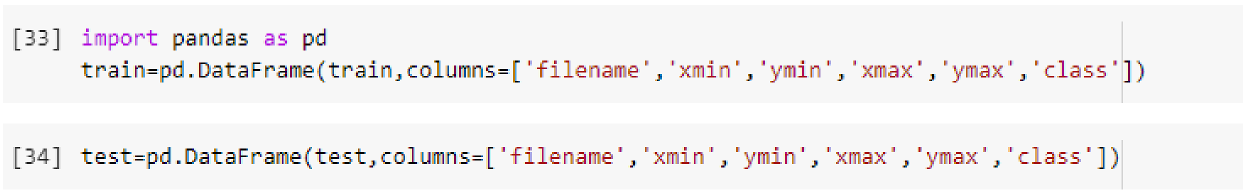
- The above one shows for the train[]

- The below one refers to the test[].



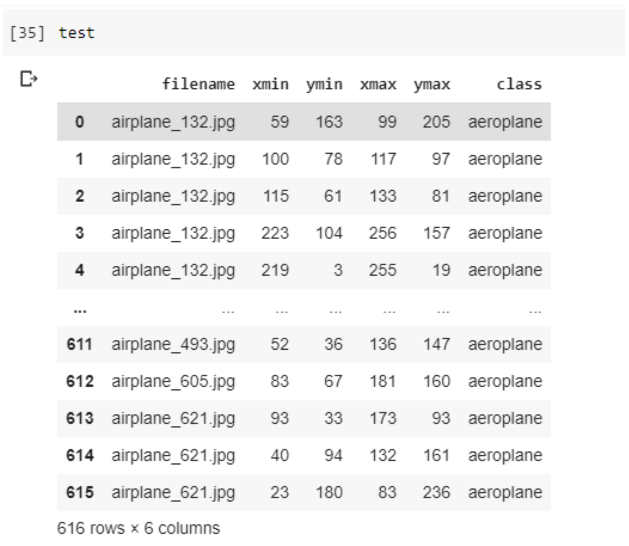
```
[29] cd -

/content/aeroplane_detection/data

[30] cd test_labels

/content/aeroplane_detection/data/test_labels

[31] !pwd

/content/aeroplane_detection/data/test_labels

[32] test=[]
     import os
     for f in os.listdir():
       file1=open(f,'r')
       for line in file1.readlines():
         line=line.strip('\n')
         l=[str(f.split('.')[0])+".jpg"]
         l=l+line.split(' ')
         l.append('aeroplane')
         test.append(l)
     print(len(test))

616
```

- Now we need to assign the column names ro the data frame for the test and train



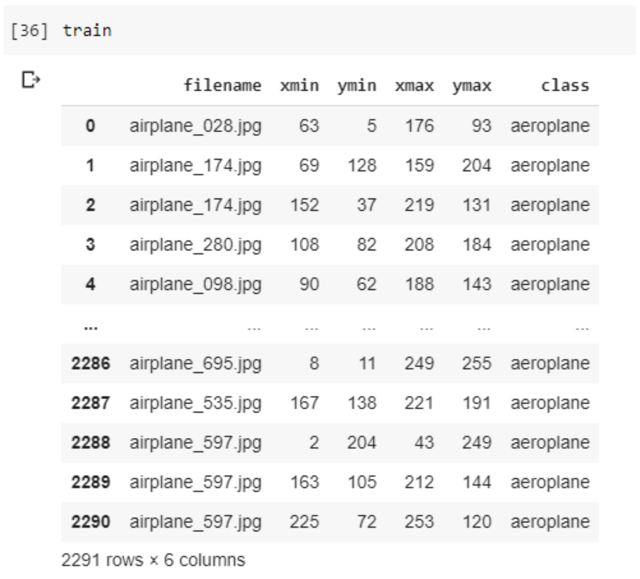
```
[33] import pandas as pd
     train=pd.DataFrame(train,columns=['filename','xmin','ymin','xmax','ymax','class'])

[34] test=pd.DataFrame(test,columns=['filename','xmin','ymin','xmax','ymax','class'])
```

- Displaying the dataframe from csv



```
[35] test

           filename  xmin  ymin  xmax  ymax      class
0    airplane_132.jpg    59   163    99   205  aeroplane
1    airplane_132.jpg   100    78   117    97  aeroplane
2    airplane_132.jpg   115    61   133    81  aeroplane
3    airplane_132.jpg   223   104   256   157  aeroplane
4    airplane_132.jpg   219     3   255    19  aeroplane
...               ...   ...   ...   ...   ...        ...
611  airplane_493.jpg    52    36   136   147  aeroplane
612  airplane_605.jpg    83    67   181   160  aeroplane
613  airplane_621.jpg    93    33   173    93  aeroplane
614  airplane_621.jpg    40    94   132   161  aeroplane
615  airplane_621.jpg    23   180    83   236  aeroplane

616 rows × 6 columns
```

- Now diplaying the train dataframe

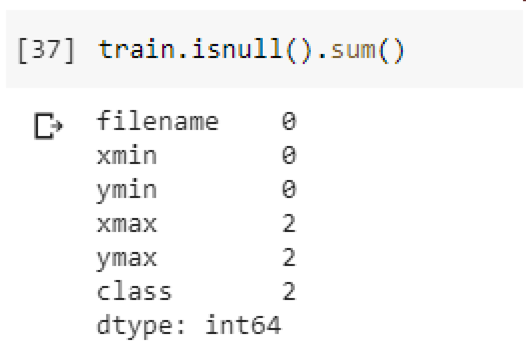


```
[36] train

            filename  xmin  ymin  xmax  ymax      class
0     airplane_028.jpg    63     5   176    93  aeroplane
1     airplane_174.jpg    69   128   159   204  aeroplane
2     airplane_174.jpg   152    37   219   131  aeroplane
3     airplane_280.jpg   108    82   208   184  aeroplane
4     airplane_098.jpg    90    62   188   143  aeroplane
...                ...   ...   ...   ...   ...        ...
2286  airplane_695.jpg     8    11   249   255  aeroplane
2287  airplane_535.jpg   167   138   221   191  aeroplane
2288  airplane_597.jpg     2   204    43   249  aeroplane
2289  airplane_597.jpg   163   105   212   144  aeroplane
2290  airplane_597.jpg   225    72   253   120  aeroplane

2291 rows × 6 columns
```

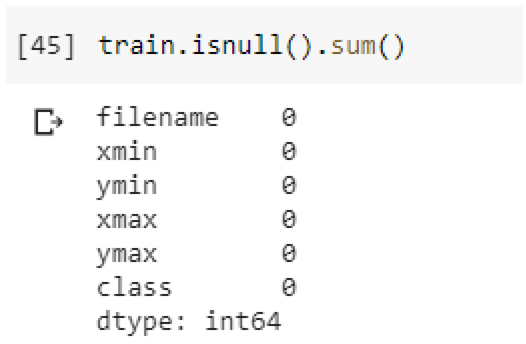
4.4.1 Identify the missing values :

- There are a number of schemes that have been developed to indicate the presence of missing data in a table or DataFrame. Generally, they revolve around one of two strategies: using a mask that globally indicates missing values, or choosing a sentinel value that indicates a missing entry.In the masking approach, the mask might be an entirely separate Boolean array, or it may involve appropriation of one bit in the data representation to locally indicate the null status of a value.In the sentinel approach, the sentinel value could be some data-specific convention, such as indicating a missing integer value with -9999 or some rare bit pattern, or it could be a more global convention, such as indicating a missing floating-point value with NaN (Not a Number), a special value which is part of the IEEE floating-point specification.

- Before converting it into the train.to_csv we need to clean the data . That means need to check the null values and check any missing values are present my in data through the command 'dataframename.isnull().sum()'. So now I need to check my train data and test data.

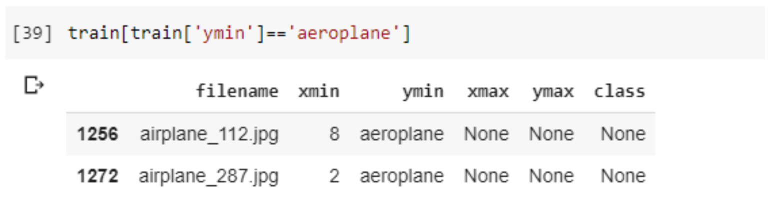


```
[37] train.isnull().sum()

filename    0
xmin        0
ymin        0
xmax        2
ymax        2
class       2
dtype: int64
```

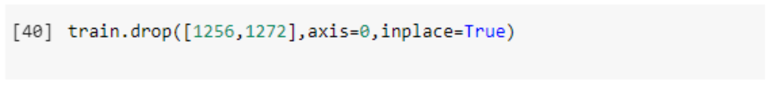


```
[45] train.isnull().sum()

filename    0
xmin        0
ymin        0
xmax        0
ymax        0
class       0
dtype: int64
```

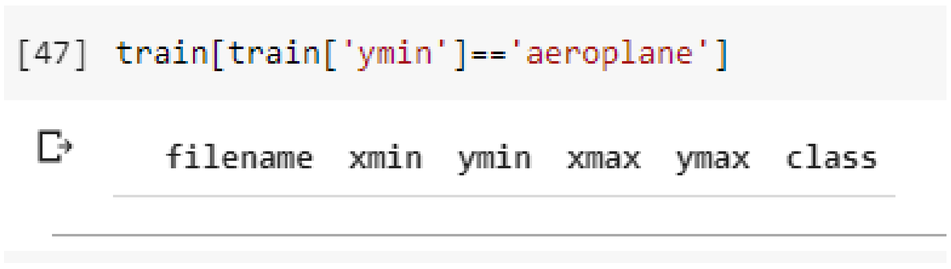
- In my train data I have two missing values , so I have to drop those rows . through the command ‘dataframename.drop([],axis=’’, inplace=True)’.



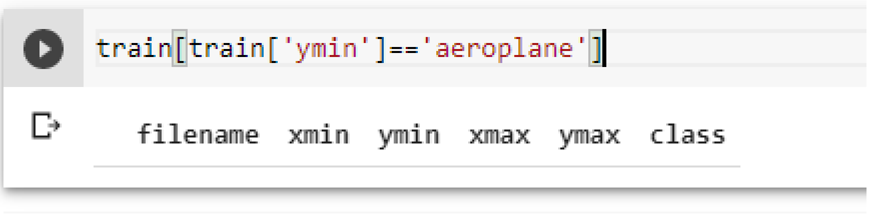
```
[39] train[train['ymin']=='aeroplane']

           filename  xmin       ymin  xmax  ymax  class
1256  airplane_112.jpg     8  aeroplane  None  None   None
1272  airplane_287.jpg     2  aeroplane  None  None   None
```



```
[40] train.drop([1256,1272],axis=0,inplace=True)
```



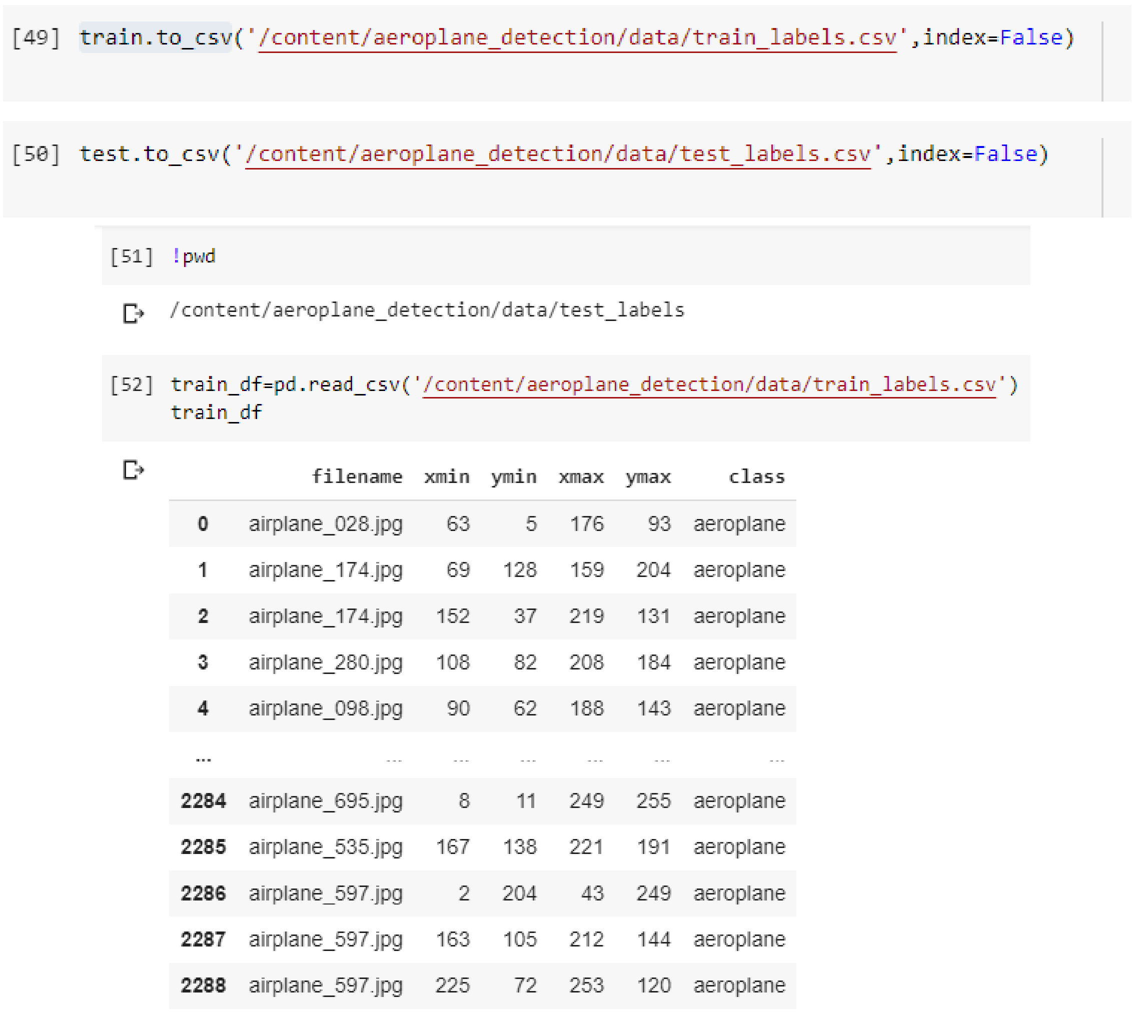
```
[47] train[train['ymin']=='aeroplane']

     filename  xmin  ymin  xmax  ymax  class
```



```
train[train['ymin']=='aeroplane']
```

```
   filename  xmin  ymin  xmax  ymax  class
```

Now we need to convert it into the train.to_csv and test.to_csv



```
[49] train.to_csv('/content/aeroplane_detection/data/train_labels.csv',index=False)
```

```
[50] test.to_csv('/content/aeroplane_detection/data/test_labels.csv',index=False)
```

```
[51] !pwd
```

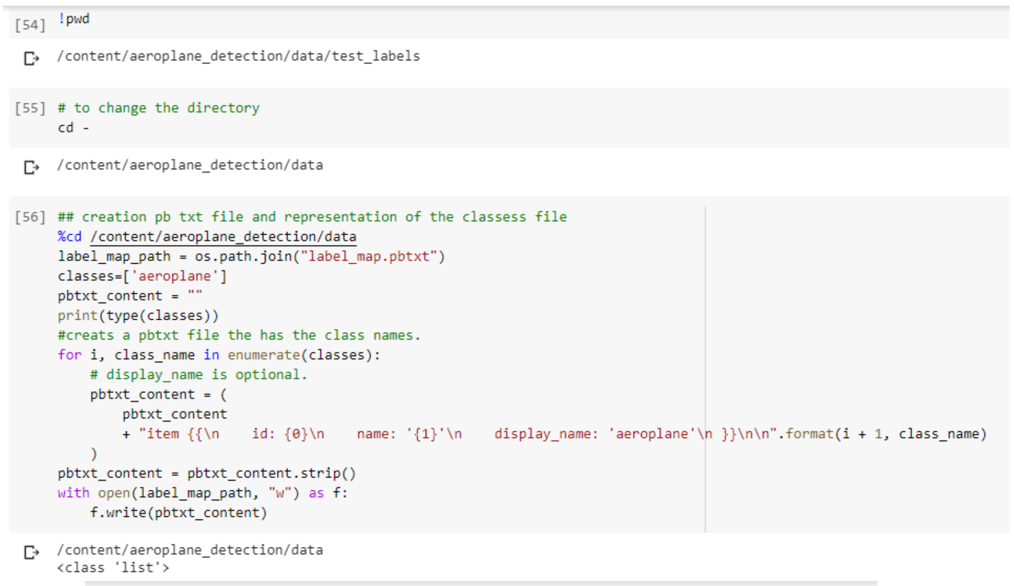
```
/content/aeroplane_detection/data/test_labels
```

```
[52] train_df=pd.read_csv('/content/aeroplane_detection/data/train_labels.csv')
     train_df
```

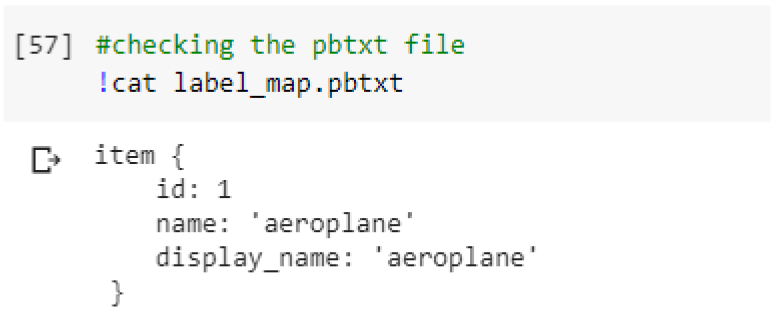
```
              filename  xmin  ymin  xmax  ymax      class
   0  airplane_028.jpg    63     5   176    93  aeroplane
   1  airplane_174.jpg    69   128   159   204  aeroplane
   2  airplane_174.jpg   152    37   219   131  aeroplane
   3  airplane_280.jpg   108    82   208   184  aeroplane
   4  airplane_098.jpg    90    62   188   143  aeroplane
 ...               ...   ...   ...   ...   ...        ...
2284  airplane_695.jpg     8    11   249   255  aeroplane
2285  airplane_535.jpg   167   138   221   191  aeroplane
2286  airplane_597.jpg     2   204    43   249  aeroplane
2287  airplane_597.jpg   163   105   212   144  aeroplane
2288  airplane_597.jpg   225    72   253   120  aeroplane
```

- Now we need to create one label_map_pbtxt file.
- **pbtxt files** in which you write flow of your pipeline with the help of input-streams, nodes and output-streams.
- The label_map .pbtxt files has been created and checked in the data folder whether

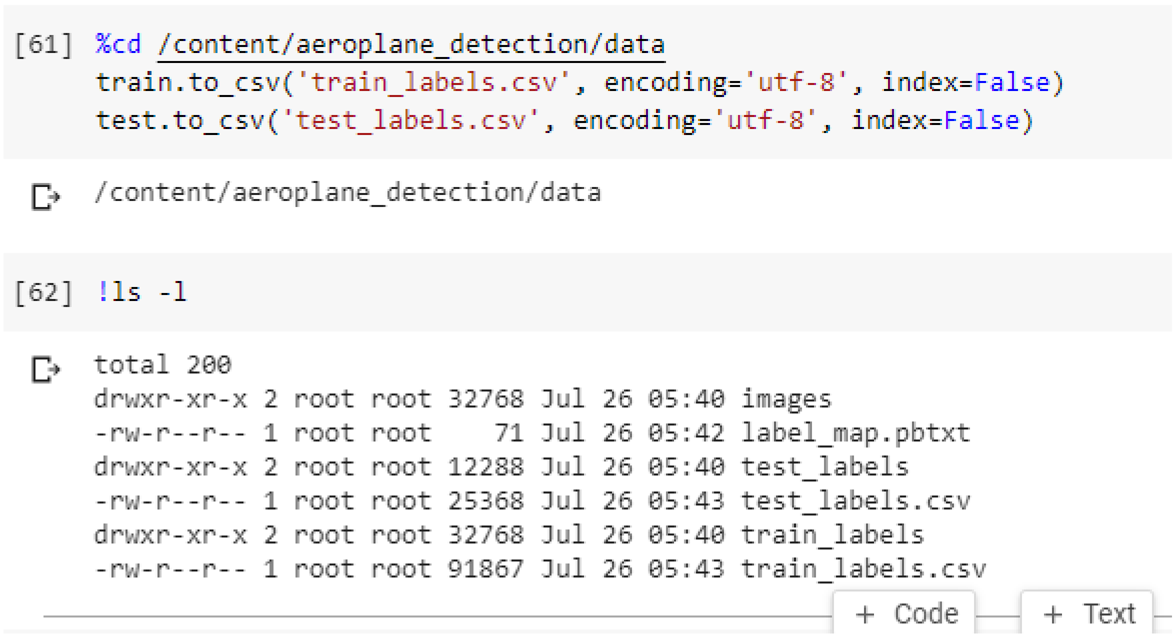
  the files are present or not.



```
[54] !pwd

/content/aeroplane_detection/data/test_labels

[55] # to change the directory
     cd -

/content/aeroplane_detection/data

[56] ## creation pb txt file and representation of the classess file
     %cd /content/aeroplane_detection/data
     label_map_path = os.path.join("label_map.pbtxt")
     classes=['aeroplane']
     pbtxt_content = ""
     print(type(classes))
     #creats a pbtxt file the has the class names.
     for i, class_name in enumerate(classes):
         # display_name is optional.
         pbtxt_content = (
             pbtxt_content
             + "item {{\n    id: {0}\n    name: '{1}'\n    display_name: 'aeroplane'\n }}\n\n".format(i + 1, class_name)
         )
     pbtxt_content = pbtxt_content.strip()
     with open(label_map_path, "w") as f:
         f.write(pbtxt_content)

/content/aeroplane_detection/data
<class 'list'>
```



```
[57] #checking the pbtxt file
     !cat label_map.pbtxt

item {
    id: 1
    name: 'aeroplane'
    display_name: 'aeroplane'
 }
```



```
[61] %cd /content/aeroplane_detection/data
     train.to_csv('train_labels.csv', encoding='utf-8', index=False)
     test.to_csv('test_labels.csv', encoding='utf-8', index=False)

     /content/aeroplane_detection/data

[62] !ls -l

     total 200
     drwxr-xr-x 2 root root 32768 Jul 26 05:40 images
     -rw-r--r-- 1 root root    71 Jul 26 05:42 label_map.pbtxt
     drwxr-xr-x 2 root root 12288 Jul 26 05:40 test_labels
     -rw-r--r-- 1 root root 25368 Jul 26 05:43 test_labels.csv
     drwxr-xr-x 2 root root 32768 Jul 26 05:40 train_labels
     -rw-r--r-- 1 root root 91867 Jul 26 05:43 train_labels.csv
```

4.5 **Downloading** Tensorflow **model:**

4.5.1 Tensorflow :

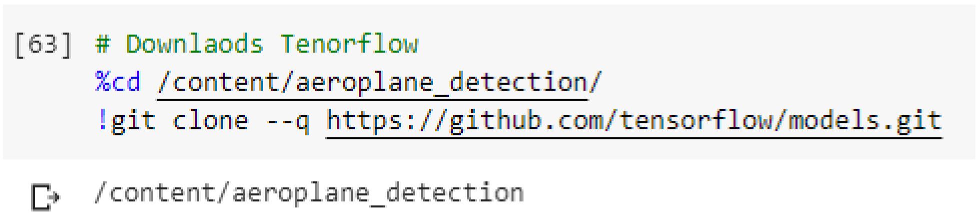
- **TensorFlow** is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

- TensorFlow is Google Brain's second-generation system. Version 1.0.0 was released on February 11, 2017. 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.

- TensorFlow offers multiple levels of abstraction so you can choose the right one for your needs. Build and train models by using the high-level Keras API, which makes getting started with TensorFlow and machine learning easy.

4.5.2. downloading the tensor flow model:

Tensorflow model contains the object detection API we are interested in. We will get it from the official repo.



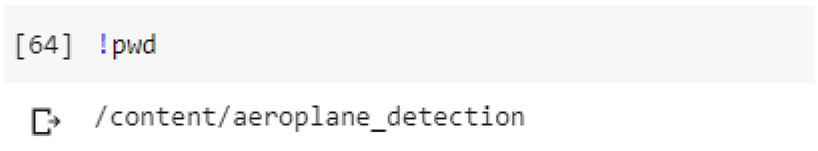
```
[63] # Downlaods Tenorflow
     %cd /content/aeroplane_detection/
     !git clone --q https://github.com/tensorflow/models.git

     /content/aeroplane_detection
```

- **git clone** is a **Git** command line utility which is used to target an existing repository and create a **clone**, or copy of the target repository. **Cloning** a local or remote repository. **Cloning** a bare repository. Using shallow options to partially **clone** repositories.

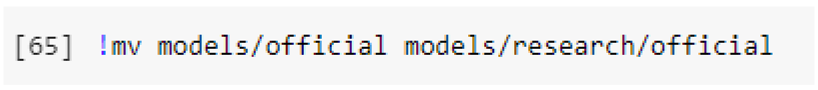
- Clones a repository into a newly created directory, creates remote-tracking branches for each branch in the cloned repository (visible using `git branch --remotes`), and creates and checks out an initial branch that is forked from the cloned repository's currently active branch.

- After the clone, a plain `git fetch` without arguments will update all the remote-tracking branches, and a `git pull` without arguments will in addition merge the remote master branch into the current master branch, if any (this is untrue when "--single-branch" is given.

- This default configuration is achieved by creating references to the remote branch heads under `refs/remotes/origin` and by initializing `remote.origin.url` and `remote.origin.fetch` Configuration variables.



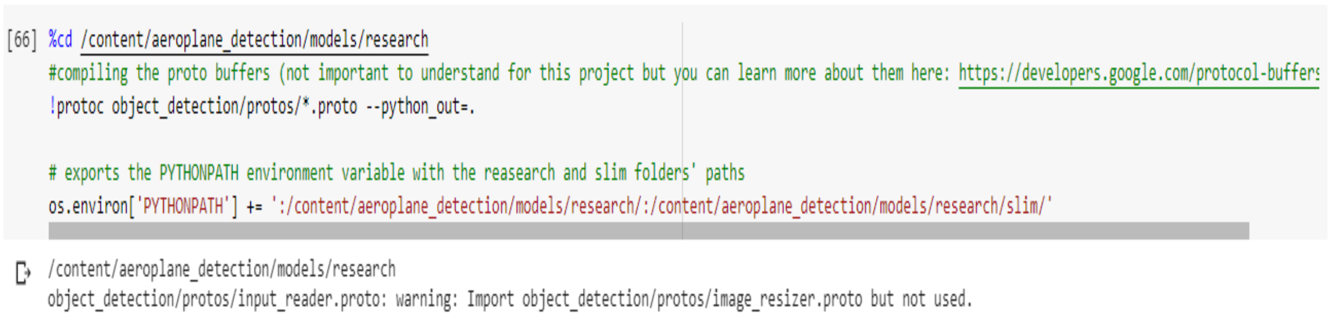
```
[64] !pwd

     /content/aeroplane_detection
```



```
[65] !mv models/official models/research/official
```

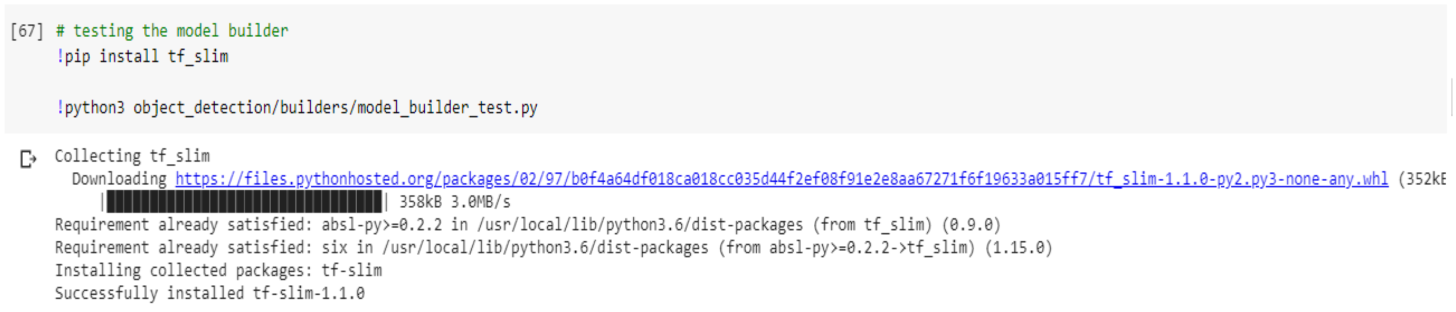
- Move command -mv . here its moving the model/official to the

Models /research/official folder



```
[66] %cd /content/aeroplane_detection/models/research
     #compiling the proto buffers (not important to understand for this project but you can learn more about them here: https://developers.google.com/protocol-buffers
     !protoc object_detection/protos/*.proto --python_out=.

     # exports the PYTHONPATH environment variable with the reasearch and slim folders' paths
     os.environ['PYTHONPATH'] += ':/content/aeroplane_detection/models/research/:/content/aeroplane_detection/models/research/slim/'

     /content/aeroplane_detection/models/research
     object_detection/protos/input_reader.proto: warning: Import object_detection/protos/image_resizer.proto but not used.
```

- Protocol buffers are Google's language-neutral, platform-neutral, extensible mechanism for serializing structured data – think XML, but smaller, faster, and simpler.
- You define how you want your data to be structured once, then you can use special generated source code to easily write and read your structured data to and from a variety of data streams and using a variety of languages.
- Protocol buffers currently support generated code in Java, Python, Objective-C, and C++.



```
[67] # testing the model builder
     !pip install tf_slim

     !python3 object_detection/builders/model_builder_test.py

     Collecting tf_slim
       Downloading https://files.pythonhosted.org/packages/02/97/b0f4a64df018ca018cc035d44f2ef08f91e2e8aa67271f6f19633a015ff7/tf_slim-1.1.0-py2.py3-none-any.whl (352kE
         |████████████████████████████████| 358kB 3.0MB/s
     Requirement already satisfied: absl-py>=0.2.2 in /usr/local/lib/python3.6/dist-packages (from tf_slim) (0.9.0)
     Requirement already satisfied: six in /usr/local/lib/python3.6/dist-packages (from absl-py>=0.2.2->tf_slim) (1.15.0)
     Installing collected packages: tf-slim
     Successfully installed tf-slim-1.1.0
```

- **ModelBuilder** is a visual programming language for building geoprocessing workflows. Geoprocessing **models** automate and document your spatial analysis and data management processes.
- **ModelBuilder** is an application you use to create, edit, and manage **models**. **Models** are workflows that string together sequences of geoprocessing tools, feeding the output of one tool into another tool as input. **ModelBuilder** can also be thought of as a visual programming language for building workflows.

- Tf_slim: Models can be succinctly **defined** using **TF-Slim** by combining its variables, layers and scopes.
- TF-Slim is a library that makes defining, training and evaluating neural networks simple:
- Allows the user to define models compactly by eliminating boilerplate code. This is accomplished through the use of argument scoping and numerous high level layers and variables. These tools increase readability and maintainability, reduce the likelihood of an error from copy-and-pasting hyperparameter values and simplifies hyperparameter tuning.
- Makes developing models simple by providing commonly used regularizers.
- Several widely used computer vision models (e.g., VGG, AlexNet) have been developed in slim, and are available to users. These can either be used as black boxes, or can be extended in various ways, e.g., by adding "multiple heads" to different internal layers.
- Slim makes it easy to extend complex models, and to warm start training algorithms by using pieces of pre-existing model checkpoints.

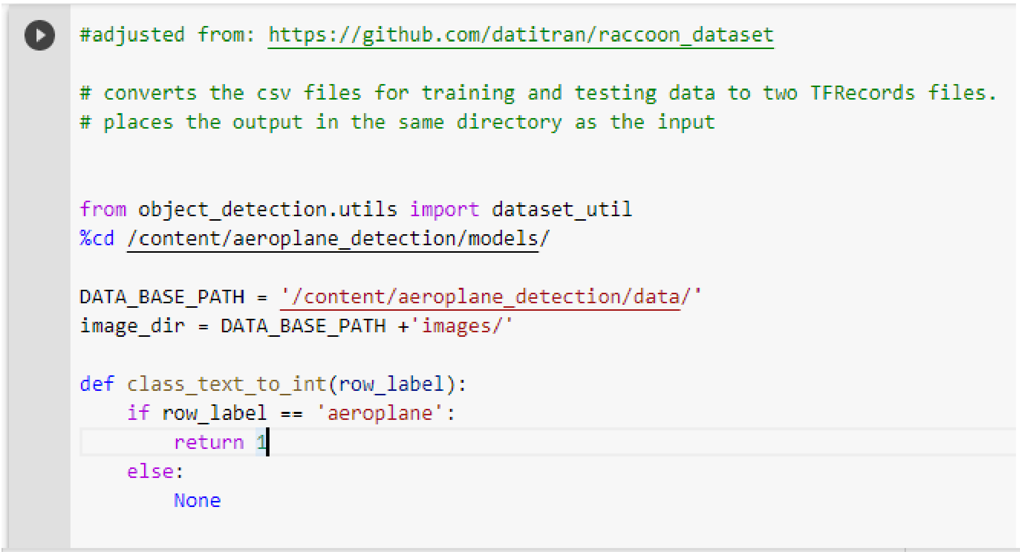
4.6 Generating TFRecords:

- We need to create the tf records for both test and train data.
- To read data efficiently it can be helpful to serialize your data and store it in a set

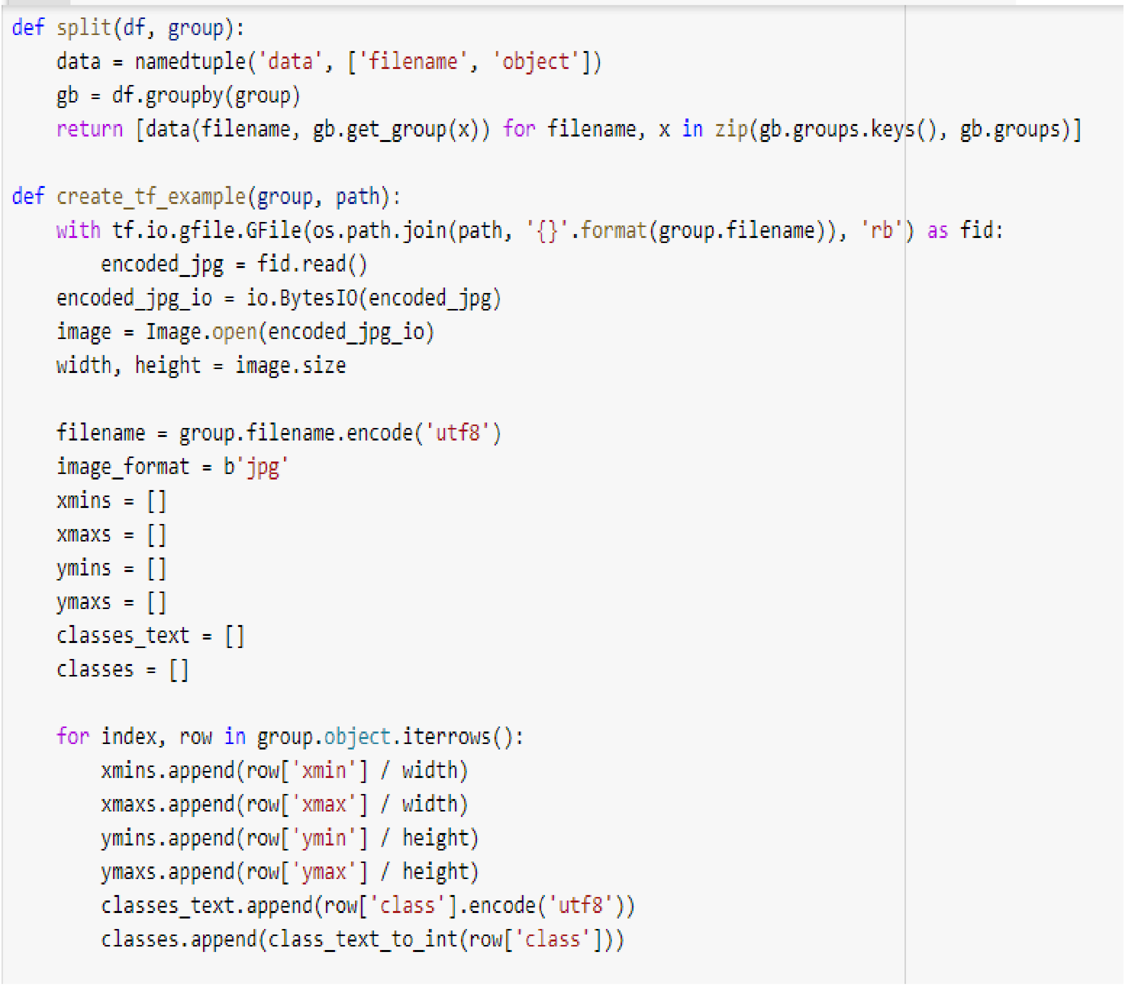
  of files (100-200MB each) that can each be read linearly. This is especially true if the data is being streamed over a network. This can also be useful for caching any data-preprocessing.
- The TFRecord format is a simple format for storing a sequence of binary records.
- Protocol buffers are a cross-platform, cross-language library for efficient serialization of structured data.
- Protocol messages are defined by .proto files, these are often the easiest way to understand a message type.

- The tf.Example message (or protobuf) is a flexible message type that represents a

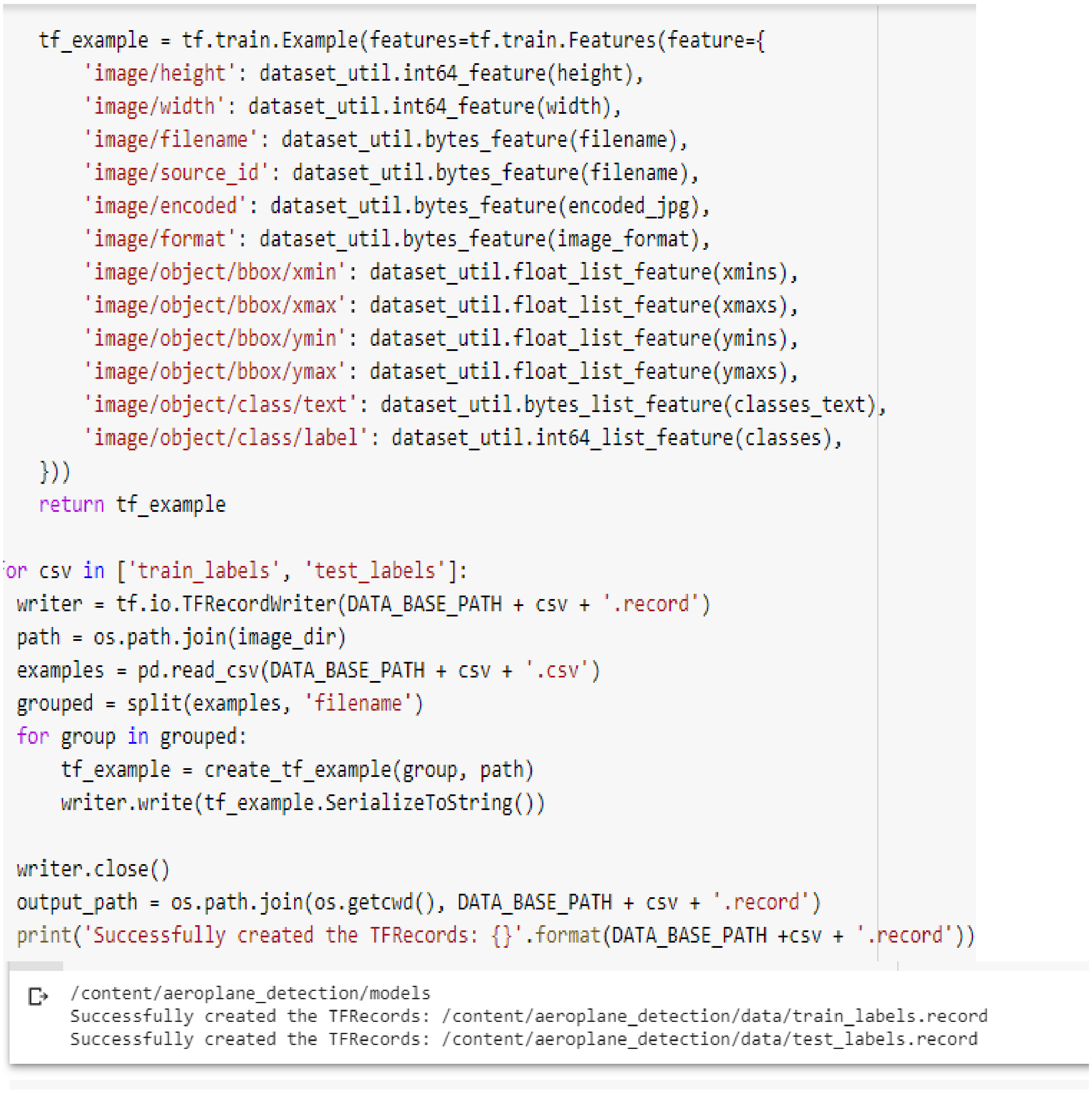
  {"string": value} mapping. It is designed for use with TensorFlow and is used throughout the higher-level APIs such as TFX.



```
#adjusted from: https://github.com/datitran/raccoon_dataset

# converts the csv files for training and testing data to two TFRecords files.
# places the output in the same directory as the input


from object_detection.utils import dataset_util
%cd /content/aeroplane_detection/models/

DATA_BASE_PATH = '/content/aeroplane_detection/data/'
image_dir = DATA_BASE_PATH +'images/'

def class_text_to_int(row_label):
    if row_label == 'aeroplane':
        return 1
    else:
        None
```



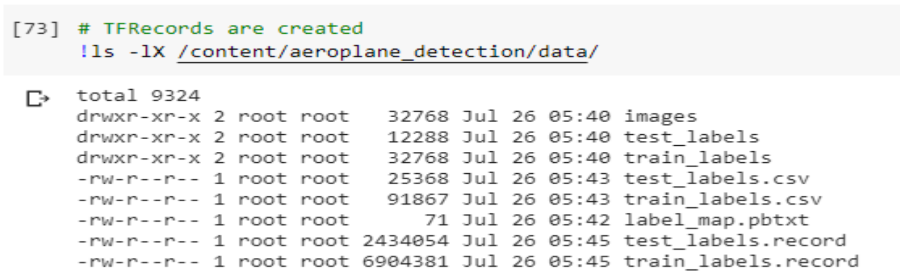
```
def split(df, group):
    data = namedtuple('data', ['filename', 'object'])
    gb = df.groupby(group)
    return [data(filename, gb.get_group(x)) for filename, x in zip(gb.groups.keys(), gb.groups)]

def create_tf_example(group, path):
    with tf.io.gfile.GFile(os.path.join(path, '{}'.format(group.filename)), 'rb') as fid:
        encoded_jpg = fid.read()
    encoded_jpg_io = io.BytesIO(encoded_jpg)
    image = Image.open(encoded_jpg_io)
    width, height = image.size

    filename = group.filename.encode('utf8')
    image_format = b'jpg'
    xmins = []
    xmaxs = []
    ymins = []
    ymaxs = []
    classes_text = []
    classes = []

    for index, row in group.object.iterrows():
        xmins.append(row['xmin'] / width)
        xmaxs.append(row['xmax'] / width)
        ymins.append(row['ymin'] / height)
        ymaxs.append(row['ymax'] / height)
        classes_text.append(row['class'].encode('utf8'))
        classes.append(class_text_to_int(row['class']))
```



```
  tf_example = tf.train.Example(features=tf.train.Features(feature={
      'image/height': dataset_util.int64_feature(height),
      'image/width': dataset_util.int64_feature(width),
      'image/filename': dataset_util.bytes_feature(filename),
      'image/source_id': dataset_util.bytes_feature(filename),
      'image/encoded': dataset_util.bytes_feature(encoded_jpg),
      'image/format': dataset_util.bytes_feature(image_format),
      'image/object/bbox/xmin': dataset_util.float_list_feature(xmins),
      'image/object/bbox/xmax': dataset_util.float_list_feature(xmaxs),
      'image/object/bbox/ymin': dataset_util.float_list_feature(ymins),
      'image/object/bbox/ymax': dataset_util.float_list_feature(ymaxs),
      'image/object/class/text': dataset_util.bytes_list_feature(classes_text),
      'image/object/class/label': dataset_util.int64_list_feature(classes),
  }))
  return tf_example

or csv in ['train_labels', 'test_labels']:
 writer = tf.io.TFRecordWriter(DATA_BASE_PATH + csv + '.record')
 path = os.path.join(image_dir)
 examples = pd.read_csv(DATA_BASE_PATH + csv + '.csv')
 grouped = split(examples, 'filename')
 for group in grouped:
     tf_example = create_tf_example(group, path)
     writer.write(tf_example.SerializeToString())

 writer.close()
 output_path = os.path.join(os.getcwd(), DATA_BASE_PATH + csv + '.record')
 print('Successfully created the TFRecords: {}'.format(DATA_BASE_PATH +csv + '.record'))
```

```
/content/aeroplane_detection/models
Successfully created the TFRecords: /content/aeroplane_detection/data/train_labels.record
Successfully created the TFRecords: /content/aeroplane_detection/data/test_labels.record
```

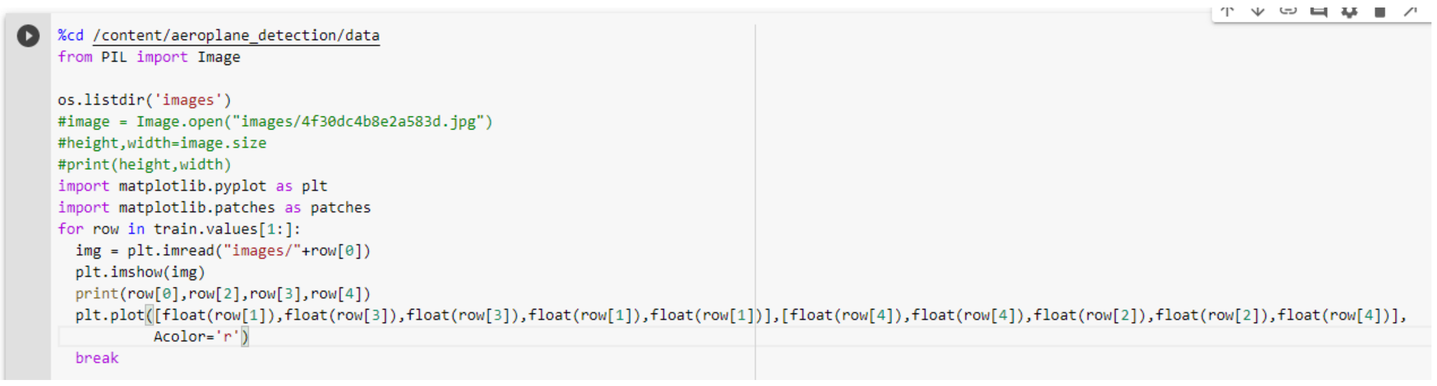
Tf records for the train_csv and test_csv files has been created.



```
[73] # TFRecords are created
     !ls -lX /content/aeroplane_detection/data/
```

```
total 9324
drwxr-xr-x 2 root root   32768 Jul 26 05:40 images
drwxr-xr-x 2 root root   12288 Jul 26 05:40 test_labels
drwxr-xr-x 2 root root   32768 Jul 26 05:40 train_labels
-rw-r--r-- 1 root root   25368 Jul 26 05:43 test_labels.csv
-rw-r--r-- 1 root root   91867 Jul 26 05:43 train_labels.csv
-rw-r--r-- 1 root root      71 Jul 26 05:42 label_map.pbtxt
-rw-r--r-- 1 root root 2434054 Jul 26 05:45 test_labels.record
-rw-r--r-- 1 root root 6904381 Jul 26 05:45 train_labels.record
```

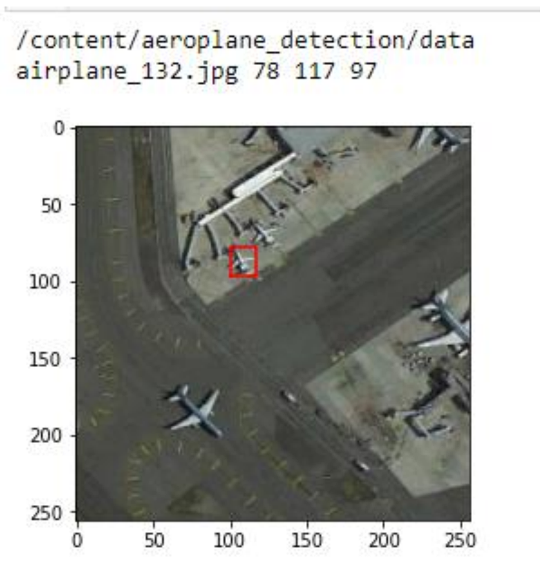
- Now we have to check wether the boundary boxes is labelling correct or not through the matplot lib.



```
%cd /content/aeroplane_detection/data
from PIL import Image

os.listdir('images')
#image = Image.open("images/4f30dc4b8e2a583d.jpg")
#height,width=image.size
#print(height,width)
import matplotlib.pyplot as plt
import matplotlib.patches as patches
for row in train.values[1:]:
  img = plt.imread("images/"+row[0])
  plt.imshow(img)
  print(row[0],row[2],row[3],row[4])
  plt.plot([float(row[1]),float(row[3]),float(row[3]),float(row[1]),float(row[1])],[float(row[4]),float(row[4]),float(row[2]),float(row[2]),float(row[4])],
           Acolor='r')
  break
```

The above is the code for the generation of the image with labels

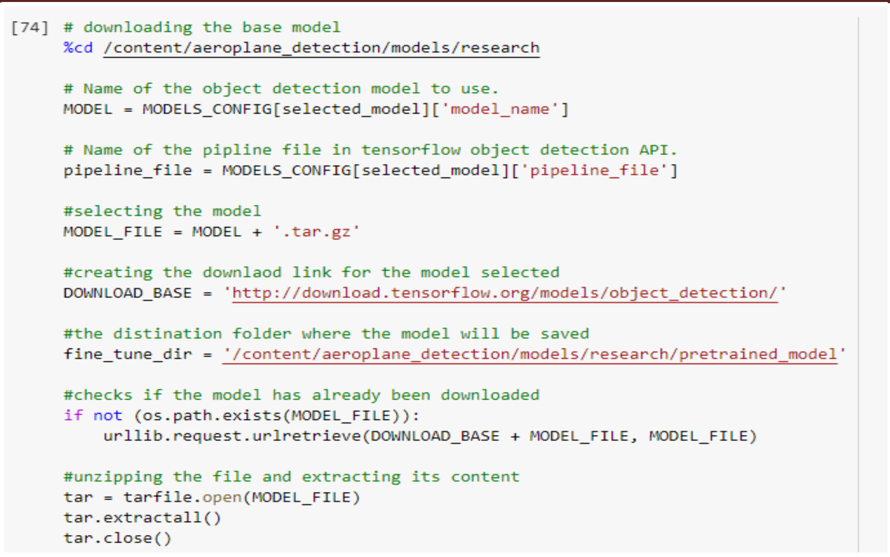
Output:



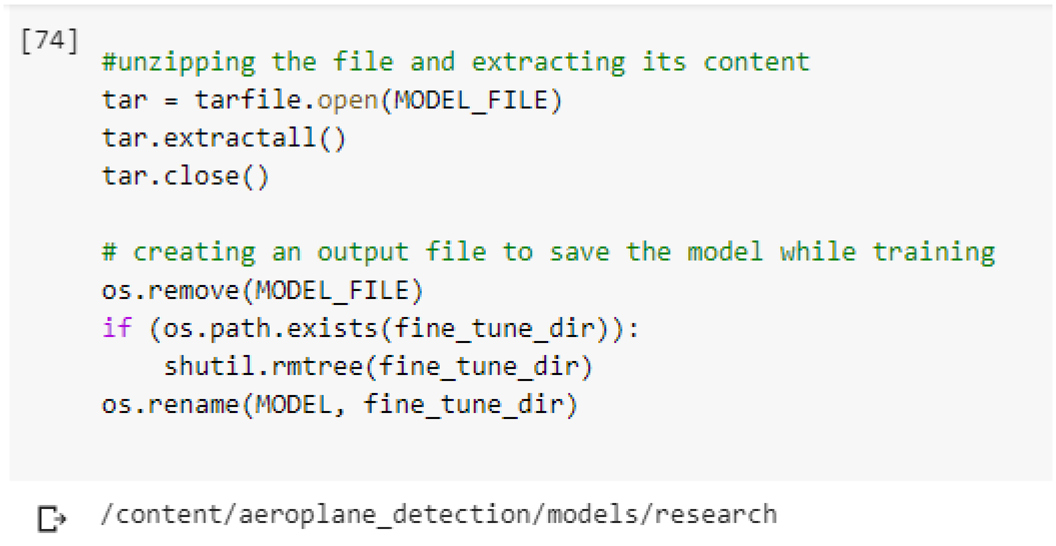
```
/content/aeroplane_detection/data
airplane_132.jpg 78 117 97
```

4.7 <u>Selecting and Downloading a Pre-trained model</u>:

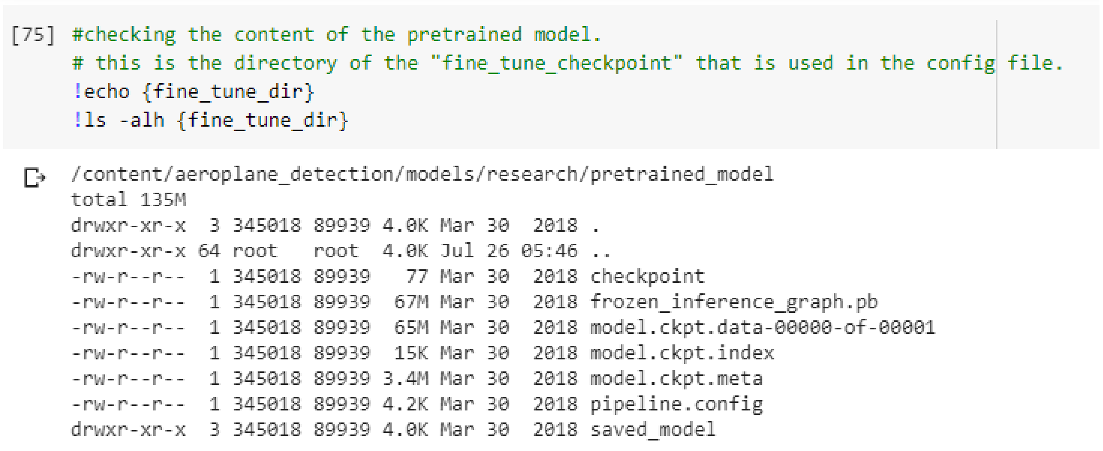
- The below code decribes the downloading of the base model that is SSD.



```
[74] # downloading the base model
     %cd /content/aeroplane_detection/models/research

     # Name of the object detection model to use.
     MODEL = MODELS_CONFIG[selected_model]['model_name']

     # Name of the pipline file in tensorflow object detection API.
     pipeline_file = MODELS_CONFIG[selected_model]['pipeline_file']

     #selecting the model
     MODEL_FILE = MODEL + '.tar.gz'

     #creating the downlaod link for the model selected
     DOWNLOAD_BASE = 'http://download.tensorflow.org/models/object_detection/'

     #the distination folder where the model will be saved
     fine_tune_dir = '/content/aeroplane_detection/models/research/pretrained_model'

     #checks if the model has already been downloaded
     if not (os.path.exists(MODEL_FILE)):
         urllib.request.urlretrieve(DOWNLOAD_BASE + MODEL_FILE, MODEL_FILE)

     #unzipping the file and extracting its content
     tar = tarfile.open(MODEL_FILE)
     tar.extractall()
     tar.close()
```



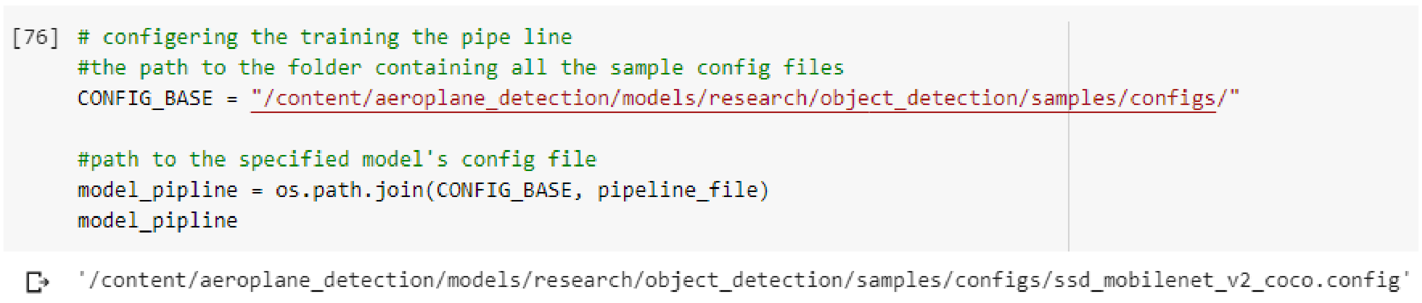
```
[74]
     #unzipping the file and extracting its content
     tar = tarfile.open(MODEL_FILE)
     tar.extractall()
     tar.close()

     # creating an output file to save the model while training
     os.remove(MODEL_FILE)
     if (os.path.exists(fine_tune_dir)):
         shutil.rmtree(fine_tune_dir)
     os.rename(MODEL, fine_tune_dir)

     /content/aeroplane_detection/models/research
```



```
[75] #checking the content of the pretrained model.
     # this is the directory of the "fine_tune_checkpoint" that is used in the config file.
     !echo {fine_tune_dir}
     !ls -alh {fine_tune_dir}

     /content/aeroplane_detection/models/research/pretrained_model
     total 135M
     drwxr-xr-x  3 345018 89939 4.0K Mar 30  2018 .
     drwxr-xr-x 64 root   root  4.0K Jul 26 05:46 ..
     -rw-r--r--  1 345018 89939   77 Mar 30  2018 checkpoint
     -rw-r--r--  1 345018 89939  67M Mar 30  2018 frozen_inference_graph.pb
     -rw-r--r--  1 345018 89939  65M Mar 30  2018 model.ckpt.data-00000-of-00001
     -rw-r--r--  1 345018 89939  15K Mar 30  2018 model.ckpt.index
     -rw-r--r--  1 345018 89939 3.4M Mar 30  2018 model.ckpt.meta
     -rw-r--r--  1 345018 89939 4.2K Mar 30  2018 pipeline.config
     drwxr-xr-x  3 345018 89939 4.0K Mar 30  2018 saved_model
```

4.8 Configuring the Training Pipeline:

- Pipeline : In computing, a **pipeline**, also known as a data **pipeline**, is a set of data processing elements connected in series, where the output of one element is the input of the next one.
- The elements of a **pipeline** are often executed in parallel or in time-sliced fashion.



```
[76] # configering the training the pipe line
     #the path to the folder containing all the sample config files
     CONFIG_BASE = "/content/aeroplane_detection/models/research/object_detection/samples/configs/"

     #path to the specified model's config file
     model_pipline = os.path.join(CONFIG_BASE, pipeline_file)
     model_pipline
```

```
'/content/aeroplane_detection/models/research/object_detection/samples/configs/ssd_mobilenet_v2_coco.config'
```

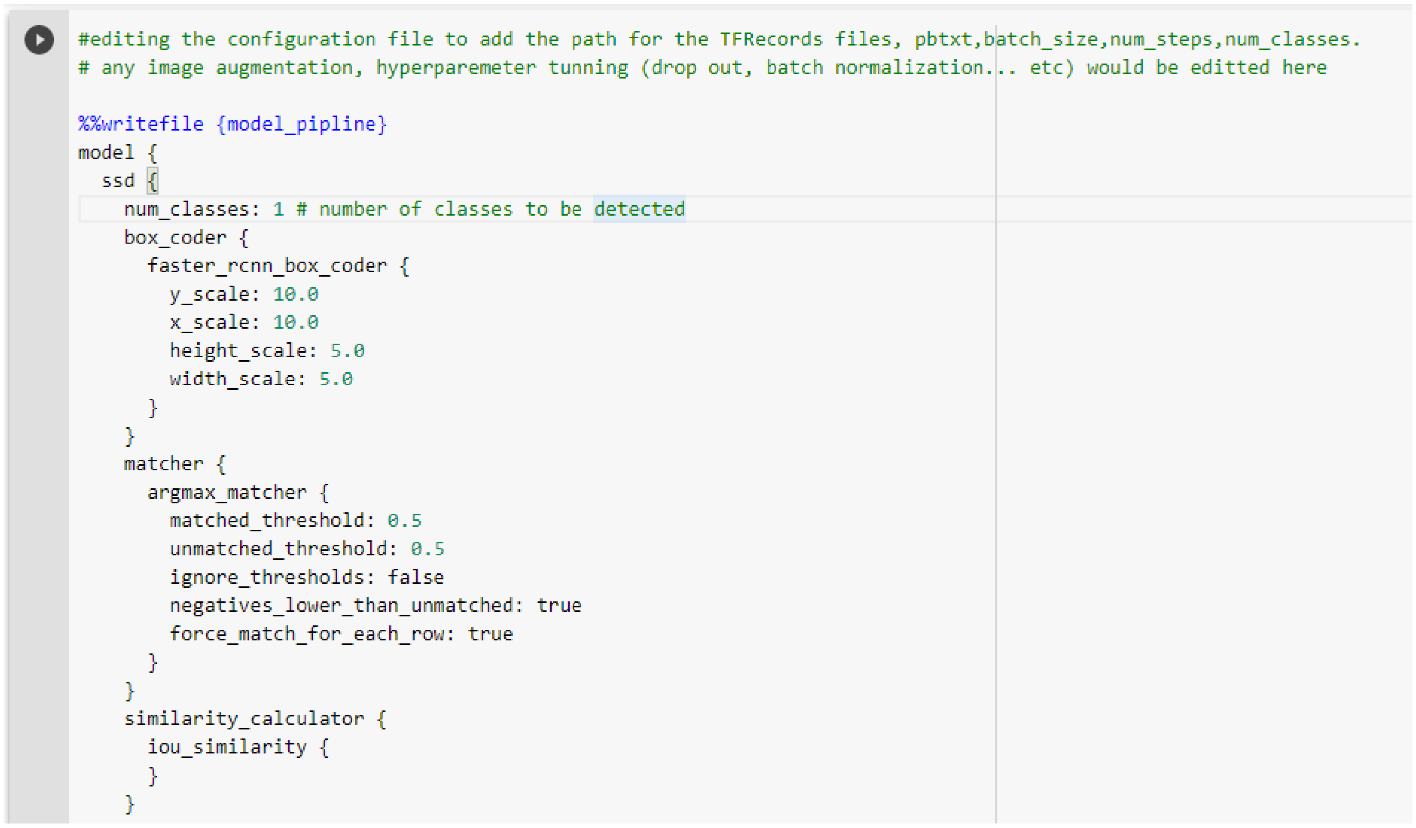
4.8.1 Basic definitions:

Encoding Categorical Data:

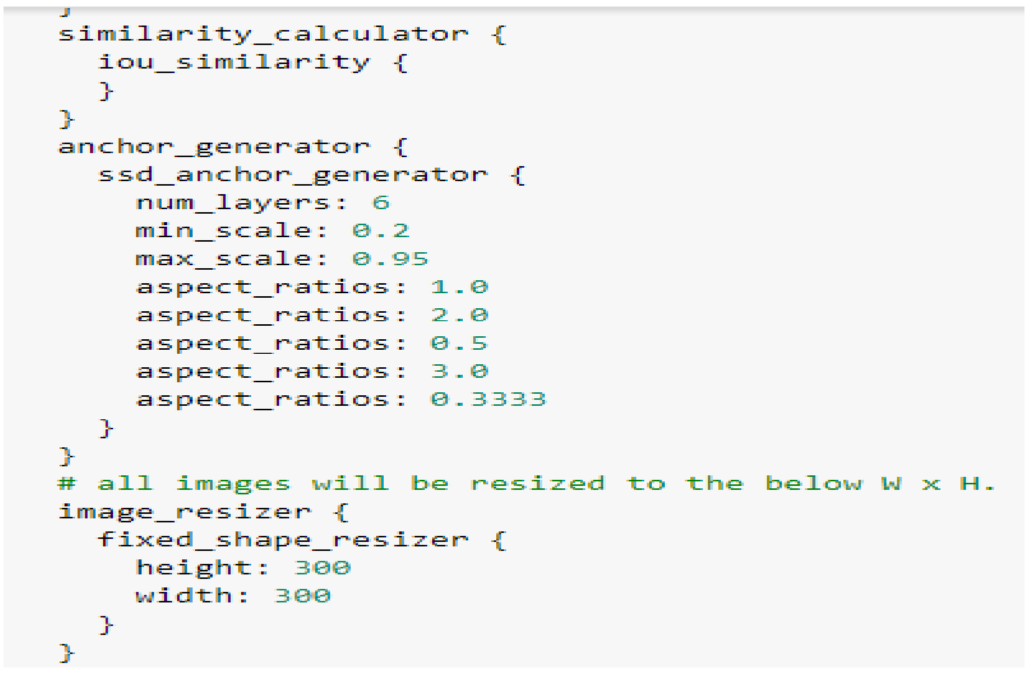
- Nominal: The categories do not have any numeric ordering in between them. They don't have any ordered relationship between each of them. Examples: Male or Female, any colour
- Ordinal: The categories have a numerical ordering in between them. Example: Graduate is less than Post Graduate, Post Graduate is less than Ph.D. customer satisfaction survey, high low medium
- Image augmentation : **Image augmentation** artificially creates training images through different ways of processing or combination of multiple processing.
- Hyperparameter tuning: it is choosing a set of optimal hyperparameters for a learning algorithm".
- Drop out : The **term** "**dropout**" refers to **dropping out** units (both hidden and visible) in a neural network. Simply put, **dropout** refers to ignoring units (i.e. neurons) during the training phase of certain set of neurons which is chosen at

random.

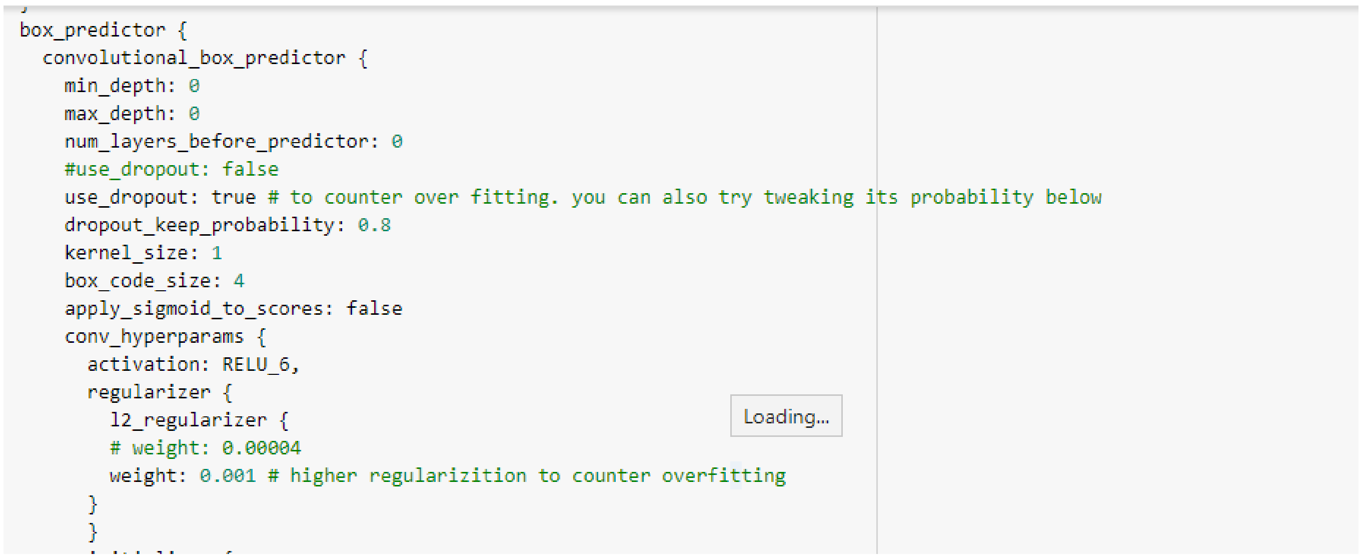
- Batch normalization : **Batch normalization** is a technique for training very deep neural networks that standardizes the inputs to a layer for each mini-**batch**.
- This has the effect of stabilizing the learning process and dramatically reducing the number of training epochs required to train deep networks.



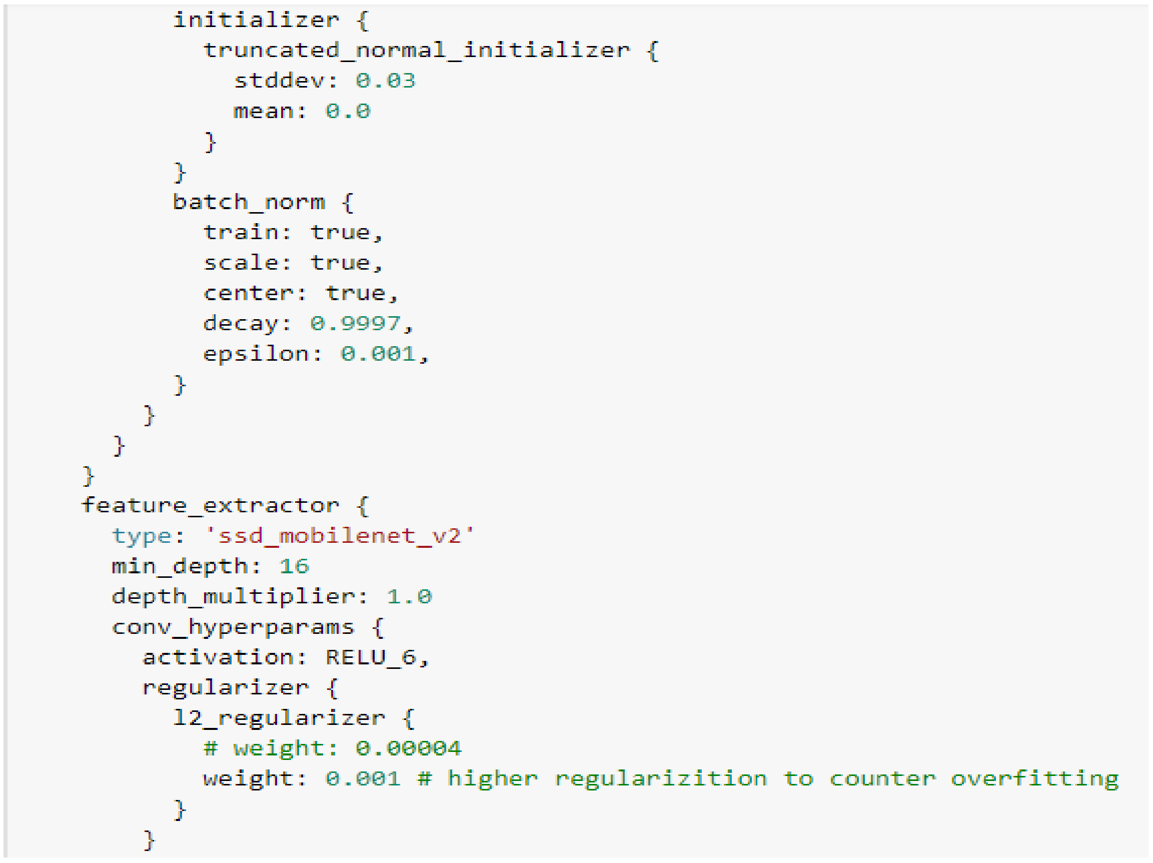
```
#editing the configuration file to add the path for the TFRecords files, pbtxt,batch_size,num_steps,num_classes.
# any image augmentation, hyperparemeter tunning (drop out, batch normalization... etc) would be editted here

%%writefile {model_pipline}
model {
  ssd {
    num_classes: 1 # number of classes to be detected
    box_coder {
      faster_rcnn_box_coder {
        y_scale: 10.0
        x_scale: 10.0
        height_scale: 5.0
        width_scale: 5.0
      }
    }
    matcher {
      argmax_matcher {
        matched_threshold: 0.5
        unmatched_threshold: 0.5
        ignore_thresholds: false
        negatives_lower_than_unmatched: true
        force_match_for_each_row: true
      }
    }
    similarity_calculator {
      iou_similarity {
      }
    }
```



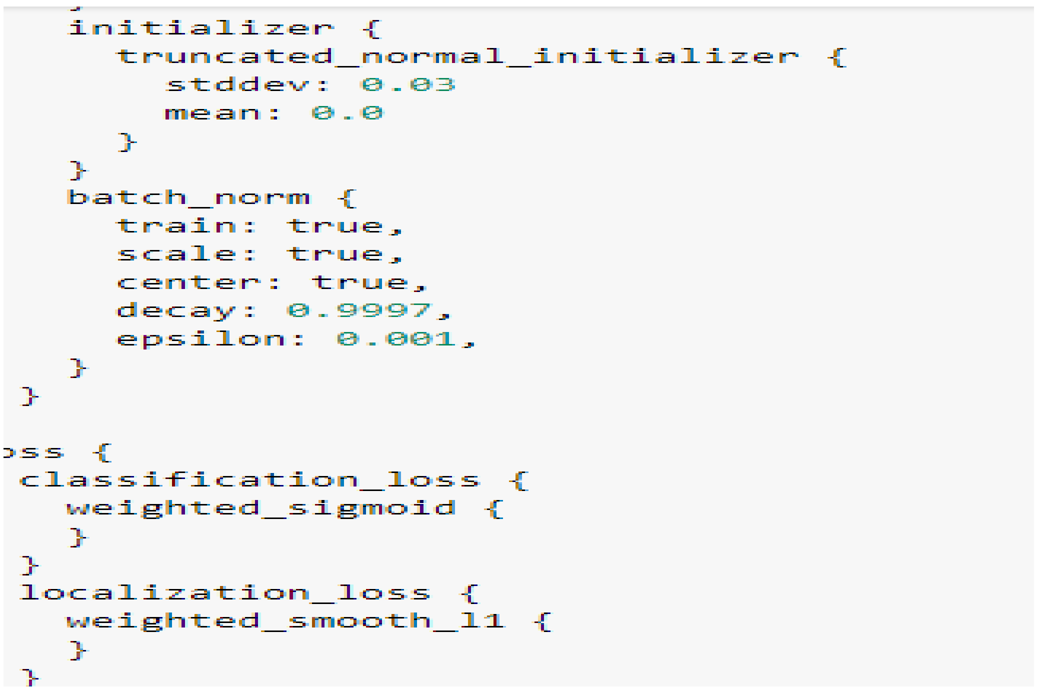
```
    similarity_calculator {
      iou_similarity {
      }
    }
    anchor_generator {
      ssd_anchor_generator {
        num_layers: 6
        min_scale: 0.2
        max_scale: 0.95
        aspect_ratios: 1.0
        aspect_ratios: 2.0
        aspect_ratios: 0.5
        aspect_ratios: 3.0
        aspect_ratios: 0.3333
      }
    }
    # all images will be resized to the below W x H.
    image_resizer {
      fixed_shape_resizer {
        height: 300
        width: 300
      }
    }
```



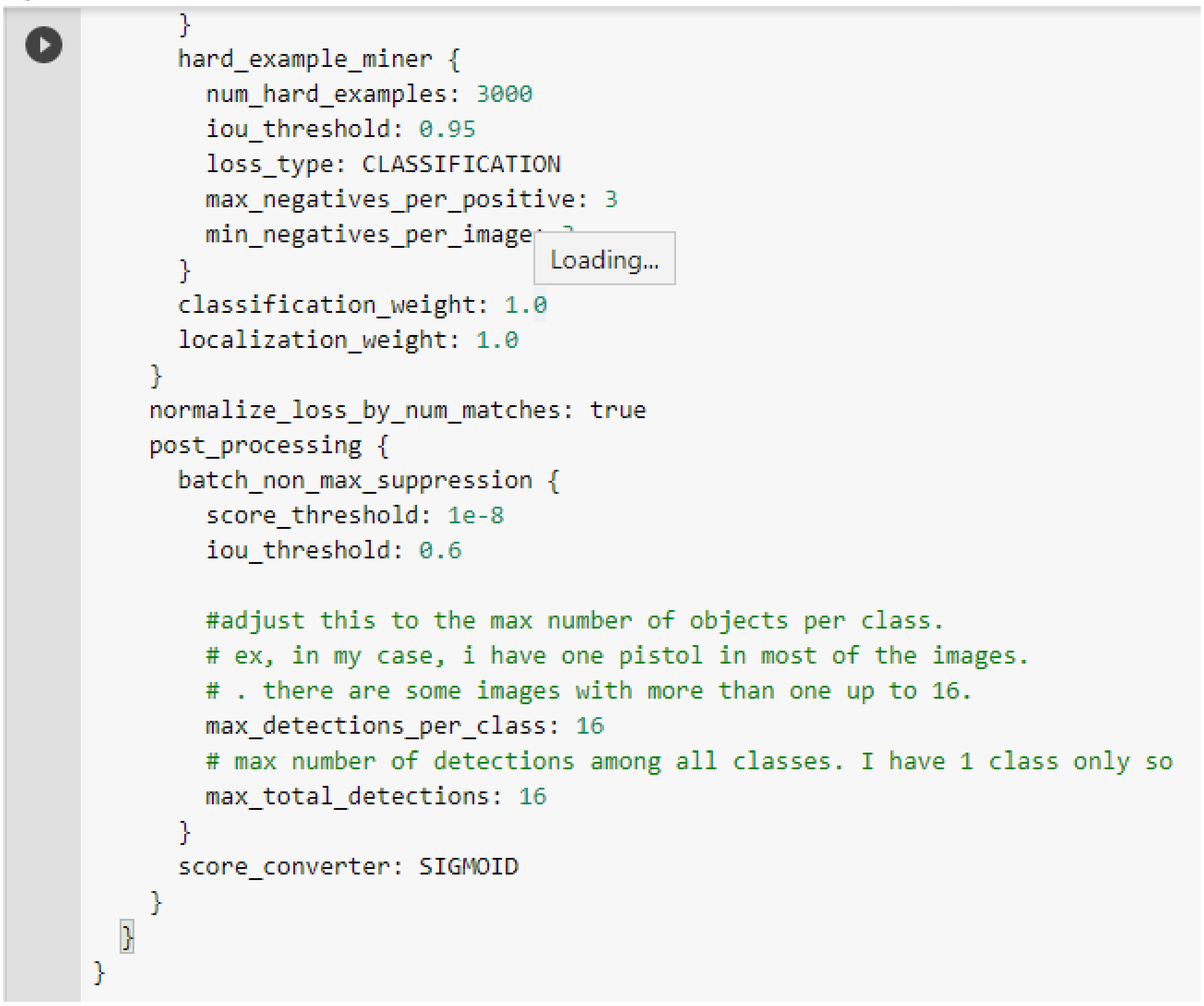
```
box_predictor {
  convolutional_box_predictor {
    min_depth: 0
    max_depth: 0
    num_layers_before_predictor: 0
    #use_dropout: false
    use_dropout: true # to counter over fitting. you can also try tweaking its probability below
    dropout_keep_probability: 0.8
    kernel_size: 1
    box_code_size: 4
    apply_sigmoid_to_scores: false
    conv_hyperparams {
      activation: RELU_6,
      regularizer {
        l2_regularizer {
        # weight: 0.00004
        weight: 0.001 # higher regularizition to counter overfitting
      }
      }
```



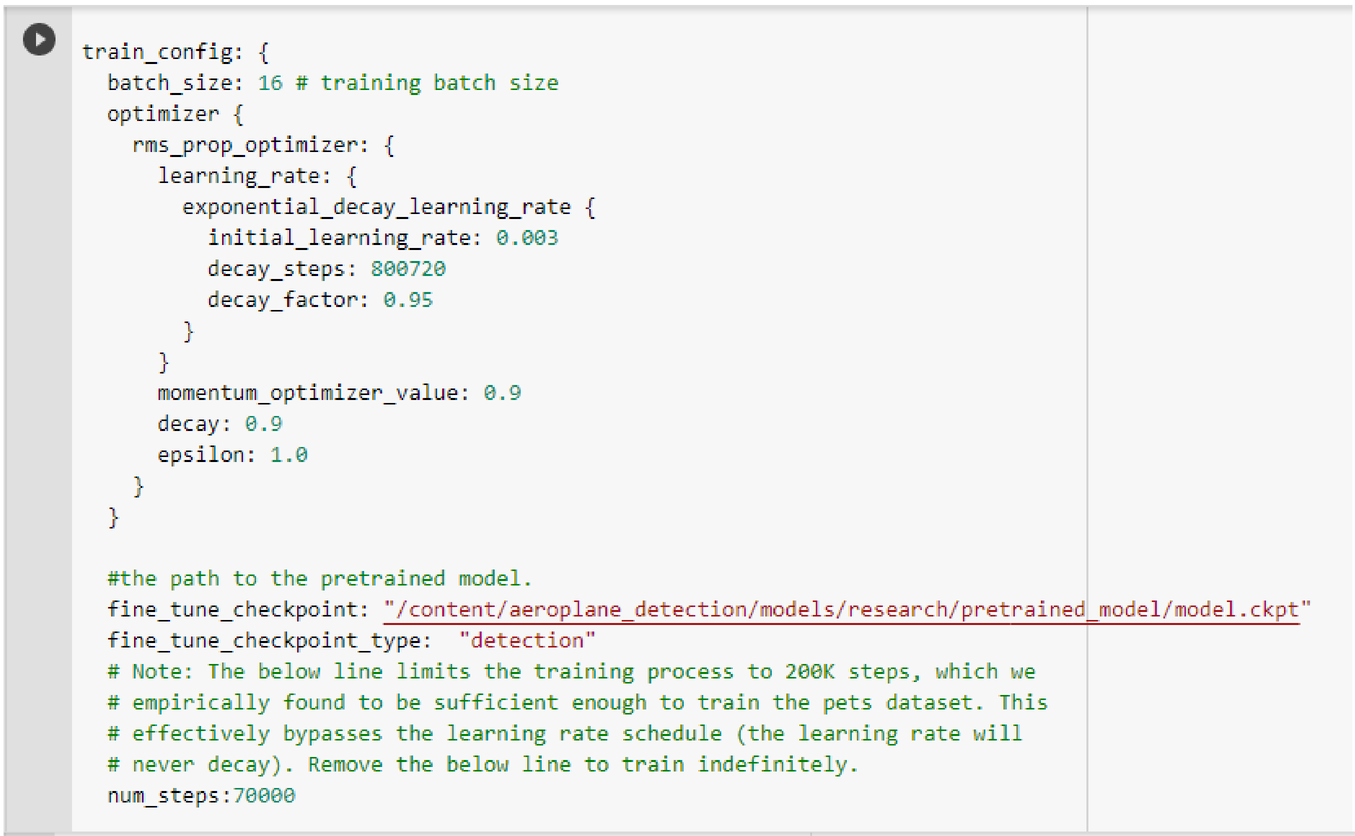
```
          initializer {
            truncated_normal_initializer {
              stddev: 0.03
              mean: 0.0
            }
          }
          batch_norm {
            train: true,
            scale: true,
            center: true,
            decay: 0.9997,
            epsilon: 0.001,
          }
        }
      }
    }
    feature_extractor {
      type: 'ssd_mobilenet_v2'
      min_depth: 16
      depth_multiplier: 1.0
      conv_hyperparams {
        activation: RELU_6,
        regularizer {
          l2_regularizer {
            # weight: 0.00004
            weight: 0.001 # higher regularizition to counter overfitting
          }
        }
```



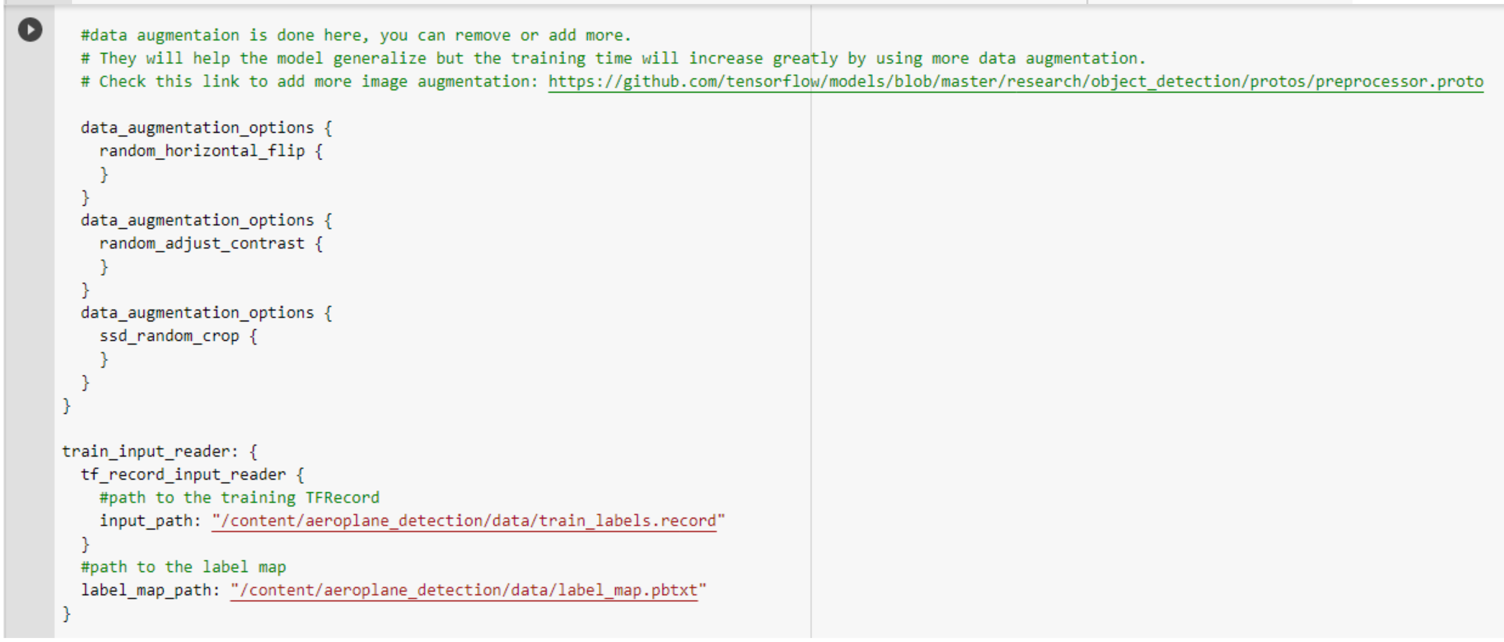
```
    initializer {
      truncated_normal_initializer {
        stddev: 0.03
        mean: 0.0
      }
    }
    batch_norm {
      train: true,
      scale: true,
      center: true,
      decay: 0.9997,
      epsilon: 0.001,
    }
  }

oss {
  classification_loss {
    weighted_sigmoid {
    }
  }
  localization_loss {
    weighted_smooth_l1 {
    }
  }
```



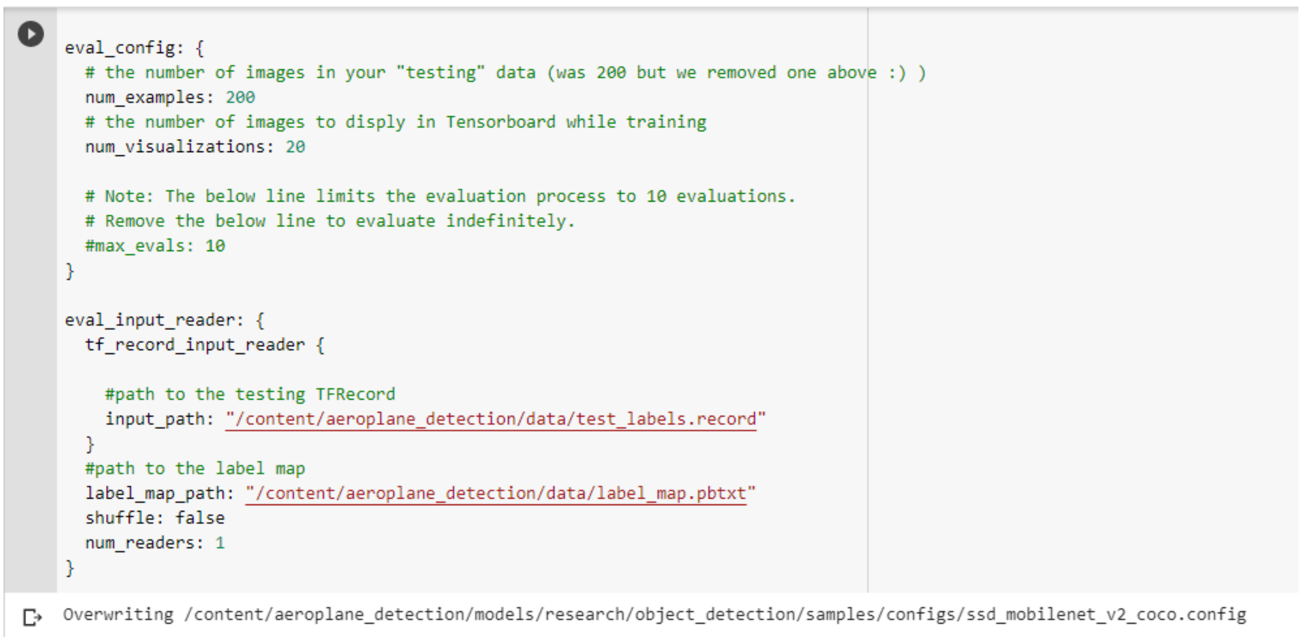
```
      }
      hard_example_miner {
        num_hard_examples: 3000
        iou_threshold: 0.95
        loss_type: CLASSIFICATION
        max_negatives_per_positive: 3
        min_negatives_per_image
      }
      classification_weight: 1.0
      localization_weight: 1.0
    }
    normalize_loss_by_num_matches: true
    post_processing {
      batch_non_max_suppression {
        score_threshold: 1e-8
        iou_threshold: 0.6

        #adjust this to the max number of objects per class.
        # ex, in my case, i have one pistol in most of the images.
        # . there are some images with more than one up to 16.
        max_detections_per_class: 16
        # max number of detections among all classes. I have 1 class only so
        max_total_detections: 16
      }
      score_converter: SIGMOID
    }
  }
}
```



```
train_config: {
  batch_size: 16 # training batch size
  optimizer {
    rms_prop_optimizer: {
      learning_rate: {
        exponential_decay_learning_rate {
          initial_learning_rate: 0.003
          decay_steps: 800720
          decay_factor: 0.95
        }
      }
      momentum_optimizer_value: 0.9
      decay: 0.9
      epsilon: 1.0
    }
  }

  #the path to the pretrained model.
  fine_tune_checkpoint: "/content/aeroplane_detection/models/research/pretrained_model/model.ckpt"
  fine_tune_checkpoint_type:  "detection"
  # Note: The below line limits the training process to 200K steps, which we
  # empirically found to be sufficient enough to train the pets dataset. This
  # effectively bypasses the learning rate schedule (the learning rate will
  # never decay). Remove the below line to train indefinitely.
  num_steps:70000
```

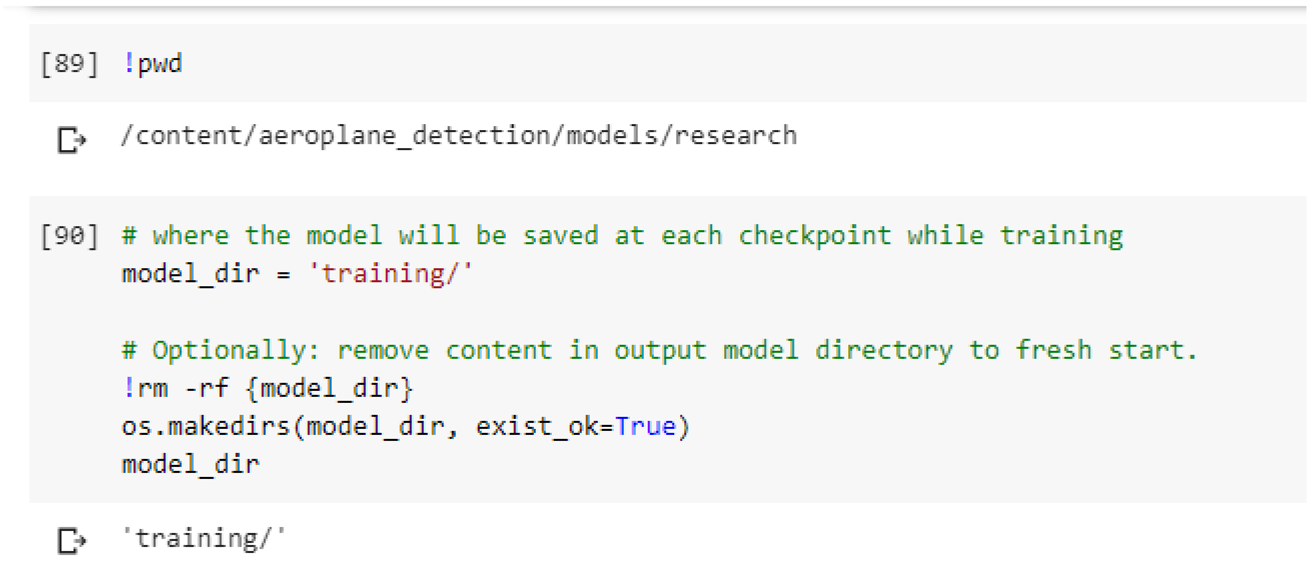


```
  #data augmentaion is done here, you can remove or add more.
  # They will help the model generalize but the training time will increase greatly by using more data augmentation.
  # Check this link to add more image augmentation: https://github.com/tensorflow/models/blob/master/research/object_detection/protos/preprocessor.proto

  data_augmentation_options {
    random_horizontal_flip {
    }
  }
  data_augmentation_options {
    random_adjust_contrast {
    }
  }
  data_augmentation_options {
    ssd_random_crop {
    }
  }
}

train_input_reader: {
  tf_record_input_reader {
    #path to the training TFRecord
    input_path: "/content/aeroplane_detection/data/train_labels.record"
  }
  #path to the label map
  label_map_path: "/content/aeroplane_detection/data/label_map.pbtxt"
}
```



```
eval_config: {
  # the number of images in your "testing" data (was 200 but we removed one above :) )
  num_examples: 200
  # the number of images to disply in Tensorboard while training
  num_visualizations: 20

  # Note: The below line limits the evaluation process to 10 evaluations.
  # Remove the below line to evaluate indefinitely.
  #max_evals: 10
}

eval_input_reader: {
  tf_record_input_reader {

    #path to the testing TFRecord
    input_path: "/content/aeroplane_detection/data/test_labels.record"
  }
  #path to the label map
  label_map_path: "/content/aeroplane_detection/data/label_map.pbtxt"
  shuffle: false
  num_readers: 1
}
```

```
Overwriting /content/aeroplane_detection/models/research/object_detection/samples/configs/ssd_mobilenet_v2_coco.config
```



```
[89] !pwd
```

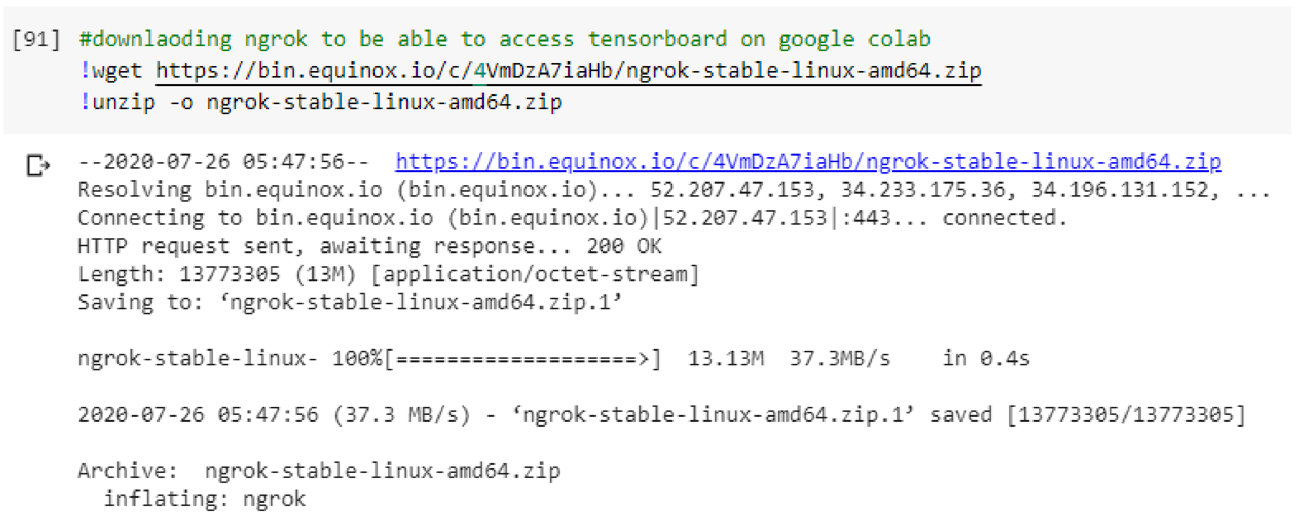
```
/content/aeroplane_detection/models/research
```

```
[90] # where the model will be saved at each checkpoint while training
     model_dir = 'training/'

     # Optionally: remove content in output model directory to fresh start.
     !rm -rf {model_dir}
     os.makedirs(model_dir, exist_ok=True)
     model_dir
```

```
'training/'
```

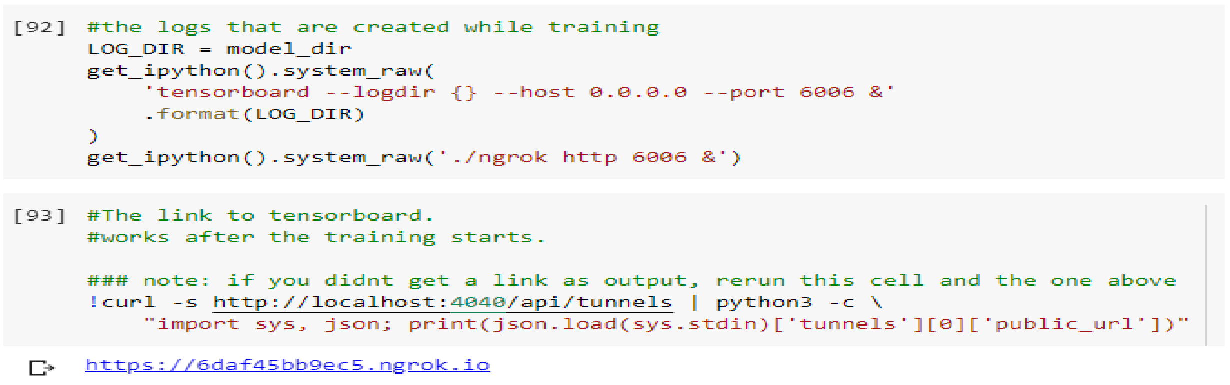
4.9 Tensor board:

- TensorBoard provides the visualization and tooling needed for machine learning experimentation
- Tracking and visualizing metrics such as loss and accuracy
- Visualizing the model graph (ops and layers)
- Viewing histograms of weights, biases, or other tensors as they change over time
- Displaying images, text, and audio data



```
[91] #downlaoding ngrok to be able to access tensorboard on google colab
     !wget https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-linux-amd64.zip
     !unzip -o ngrok-stable-linux-amd64.zip
```

```
--2020-07-26 05:47:56--  https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-linux-amd64.zip
Resolving bin.equinox.io (bin.equinox.io)... 52.207.47.153, 34.233.175.36, 34.196.131.152, ...
Connecting to bin.equinox.io (bin.equinox.io)|52.207.47.153|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 13773305 (13M) [application/octet-stream]
Saving to: ‘ngrok-stable-linux-amd64.zip.1’

ngrok-stable-linux- 100%[===================>]  13.13M  37.3MB/s    in 0.4s

2020-07-26 05:47:56 (37.3 MB/s) - ‘ngrok-stable-linux-amd64.zip.1’ saved [13773305/13773305]

Archive:  ngrok-stable-linux-amd64.zip
  inflating: ngrok
```



```
[92] #the logs that are created while training
     LOG_DIR = model_dir
     get_ipython().system_raw(
         'tensorboard --logdir {} --host 0.0.0.0 --port 6006 &'
         .format(LOG_DIR)
     )
     get_ipython().system_raw('./ngrok http 6006 &')
```

```
[93] #The link to tensorboard.
     #works after the training starts.

     ### note: if you didnt get a link as output, rerun this cell and the one above
     !curl -s http://localhost:4040/api/tunnels | python3 -c \
         "import sys, json; print(json.load(sys.stdin)['tunnels'][0]['public_url'])"
```

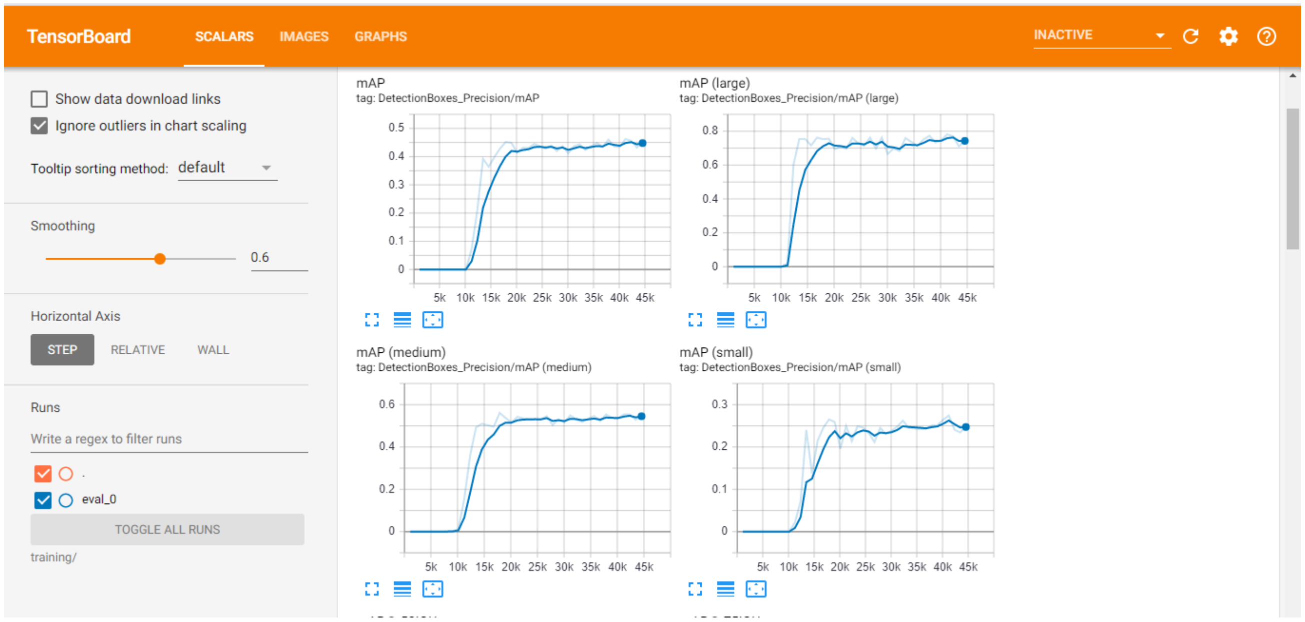
```
https://6daf45bb9ec5.ngrok.io
```

```
[94] model_pipline

'/content/aeroplane_detection/models/research/object_detection/samples/configs/ssd_mobilenet_v2_coco.config'
```

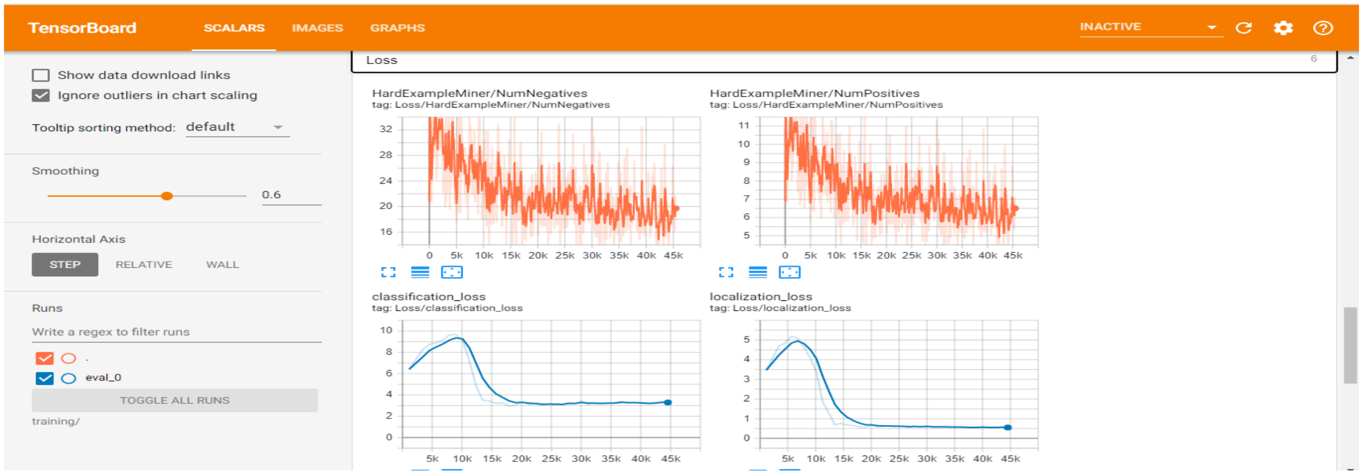
```
[95] model_dir

'training/'
```

```
[96] !pwd

/content/aeroplane_detection/models/research
```

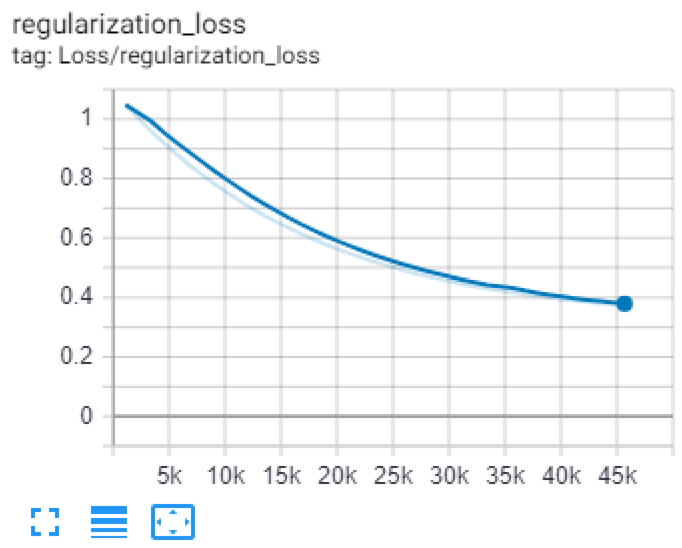
The below graphs are shown in tensor board which trained below 50000 steps.

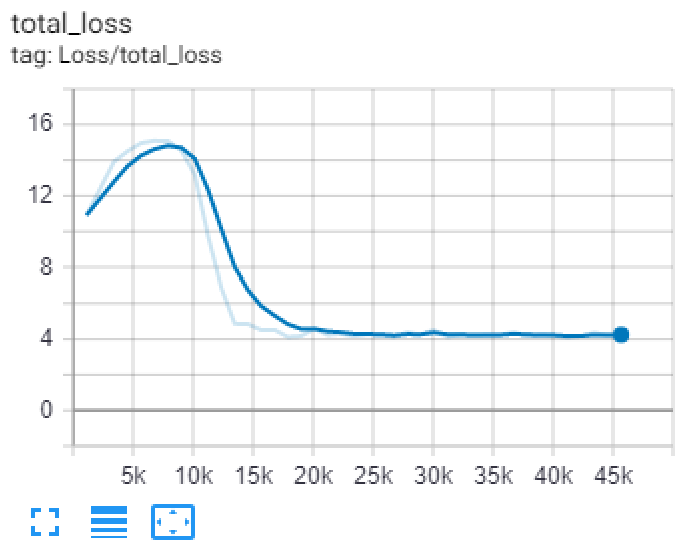

TensorBoard
SCALARS
IMAGES
GRAPHS
INACTIVE
mAP
tag: DetectionBoxes_Precision/mAP
mAP (large)
tag: DetectionBoxes_Precision/mAP (large)
mAP (medium)
tag: DetectionBoxes_Precision/mAP (medium)
mAP (small)
tag: DetectionBoxes_Precision/mAP (small)
5k 10k 15k 20k 25k 30k 35k 40k 45k
TOGGLE ALL RUNS

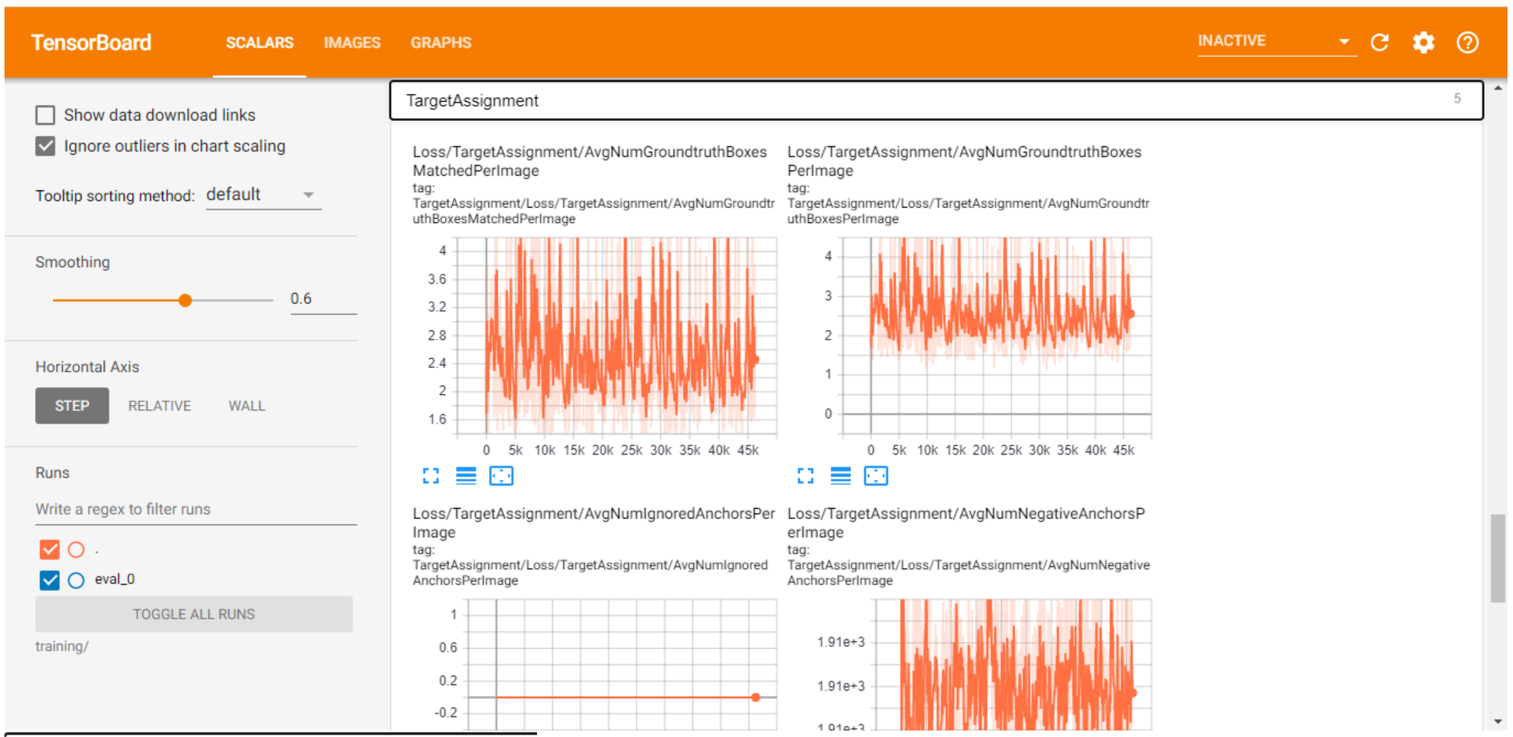
☐ Show data download links
☑ Ignore outliers in chart scaling
Tooltip sorting method: default
Smoothing 0.6
Horizontal Axis: STEP RELATIVE WALL
Runs
Write a regex to filter runs
☑ .
☑ eval_0
training/


TensorBoard
SCALARS
IMAGES
GRAPHS
INACTIVE
Loss
HardExampleMiner/NumNegatives
tag: Loss/HardExampleMiner/NumNegatives
HardExampleMiner/NumPositives
tag: Loss/HardExampleMiner/NumPositives
classification_loss
tag: Loss/classification_loss
localization_loss
tag: Loss/localization_loss
0 5k 10k 15k 20k 25k 30k 35k 40k 45k
TOGGLE ALL RUNS

☐ Show data download links
☑ Ignore outliers in chart scaling
Tooltip sorting method: default
Smoothing 0.6
Horizontal Axis: STEP RELATIVE WALL
Runs
Write a regex to filter runs
☑ .
☑ eval_0
training/







☐ Show data download links
☑ Ignore outliers in chart scaling
Tooltip sorting method: default

Smoothing 0.6

Horizontal Axis
STEP RELATIVE WALL

Runs
Write a regex to filter runs
☑ ○ .
☑ ○ eval_0
TOGGLE ALL RUNS
training/

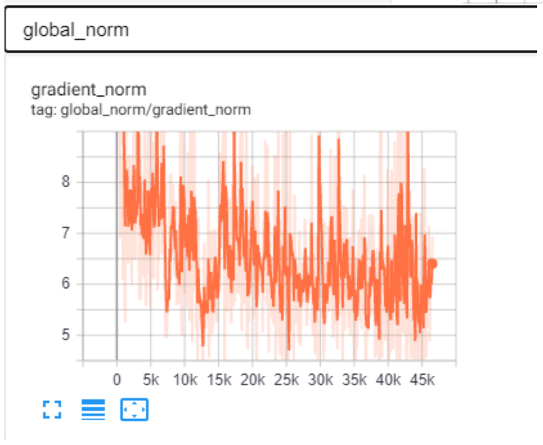
TargetAssignment 5

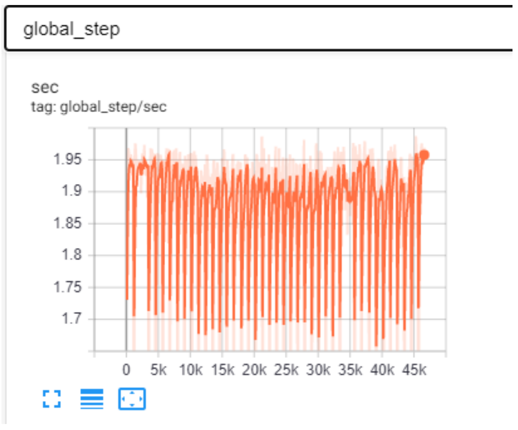
Loss/TargetAssignment/AvgNumGroundtruthBoxesMatchedPerImage
tag: TargetAssignment/Loss/TargetAssignment/AvgNumGroundtruthBoxesMatchedPerImage

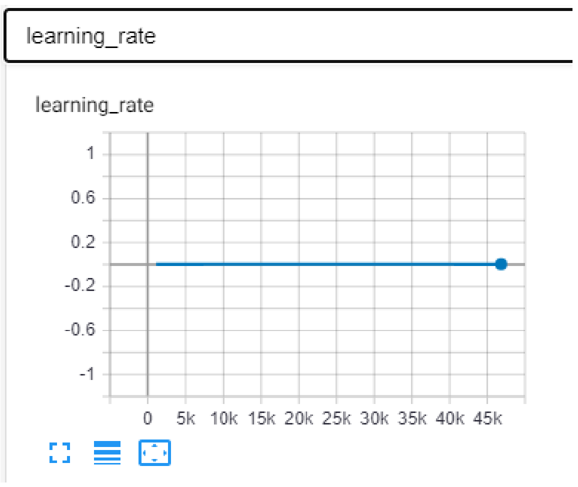
Loss/TargetAssignment/AvgNumGroundtruthBoxesPerImage
tag: TargetAssignment/Loss/TargetAssignment/AvgNumGroundtruthBoxesPerImage

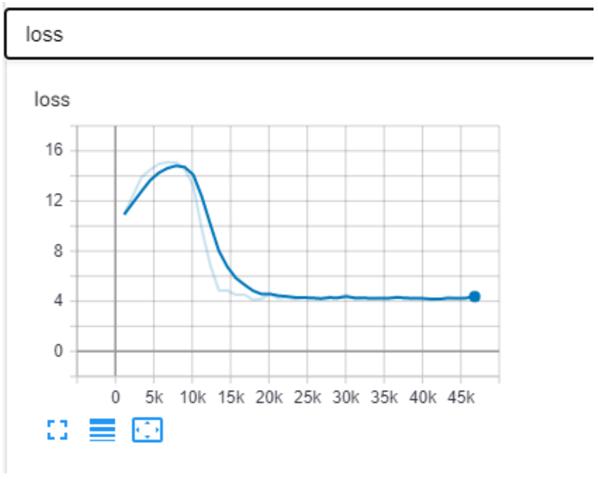
Loss/TargetAssignment/AvgNumIgnoredAnchorsPerImage
tag: TargetAssignment/Loss/TargetAssignment/AvgNumIgnoredAnchorsPerImage

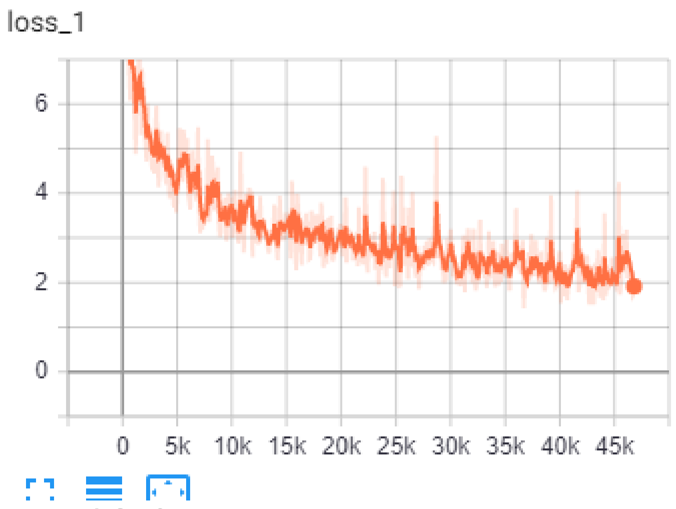
Loss/TargetAssignment/AvgNumNegativeAnchorsPerImage
tag: TargetAssignment/Loss/TargetAssignment/AvgNumNegativeAnchorsPerImage

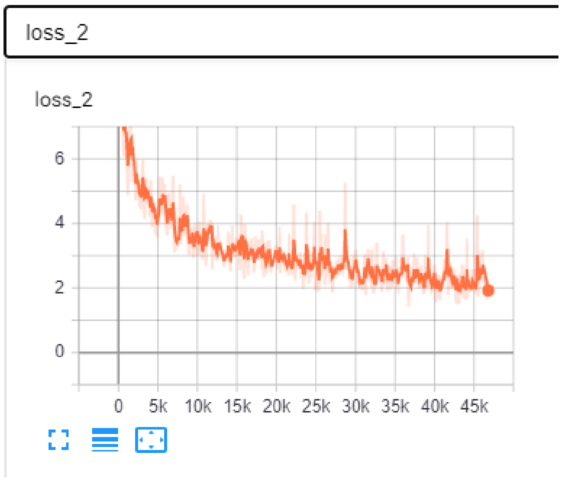




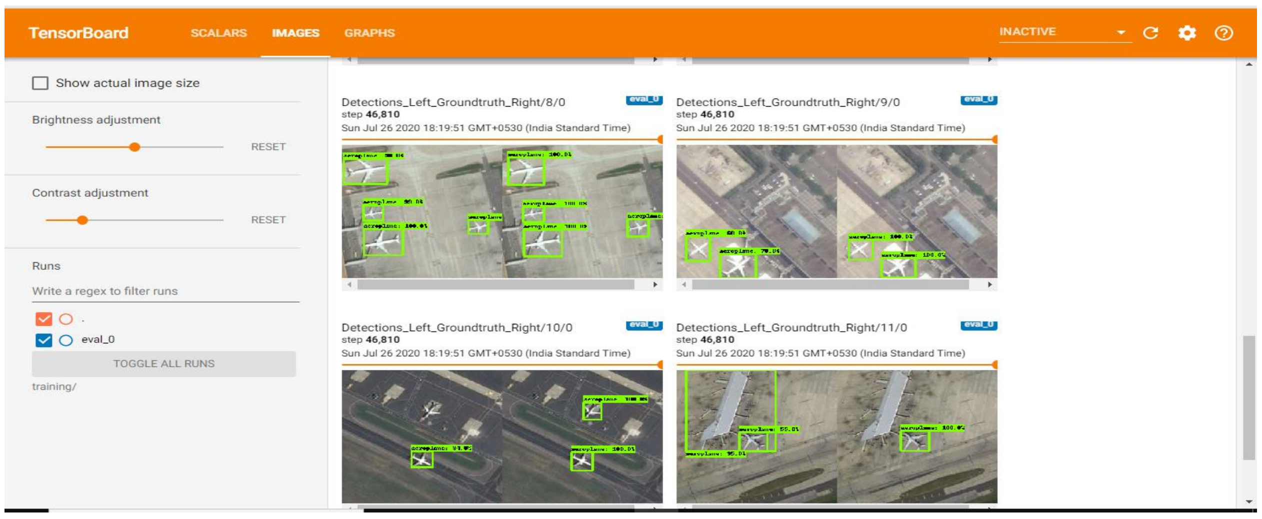








IMAGES:

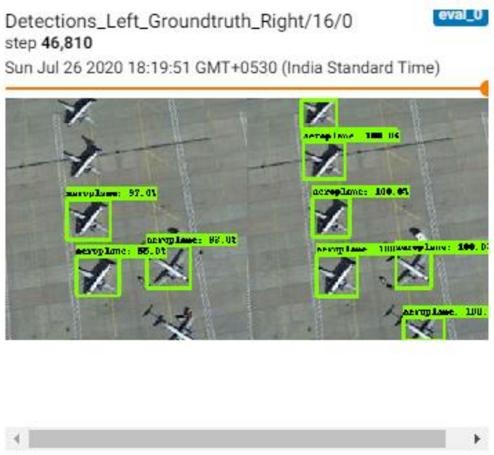


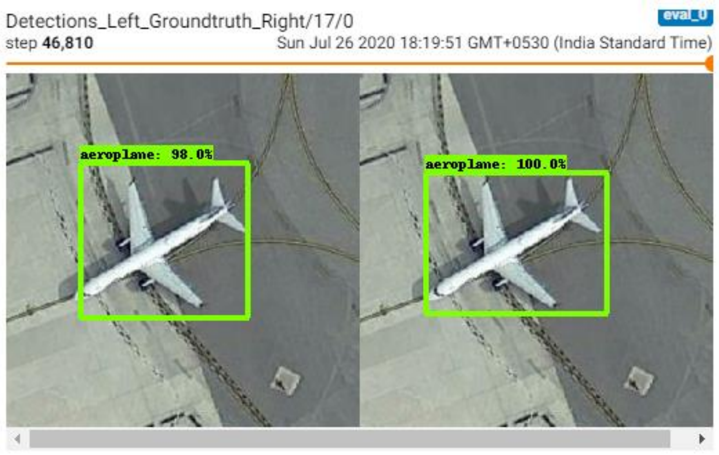
Detections_Left_Groundtruth_Right/8/0 eval_0
step 46,810
Sun Jul 26 2020 18:19:51 GMT+0530 (India Standard Time)

Detections_Left_Groundtruth_Right/9/0 eval_0
step 46,810
Sun Jul 26 2020 18:19:51 GMT+0530 (India Standard Time)

Detections_Left_Groundtruth_Right/10/0 eval_0
step 46,810
Sun Jul 26 2020 18:19:51 GMT+0530 (India Standard Time)

Detections_Left_Groundtruth_Right/11/0 eval_0
step 46,810
Sun Jul 26 2020 18:19:51 GMT+0530 (India Standard Time)

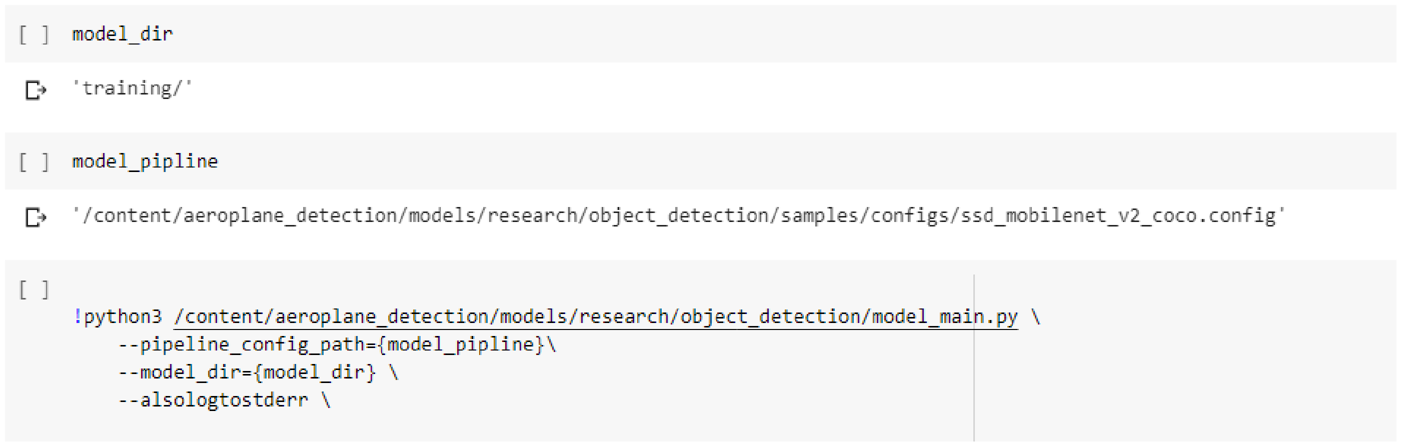

Detections_Left_Groundtruth_Right/16/0
eval_0
step 46,810
Sun Jul 26 2020 18:19:51 GMT+0530 (India Standard Time)


Detections_Left_Groundtruth_Right/17/0
eval_0
step 46,810
Sun Jul 26 2020 18:19:51 GMT+0530 (India Standard Time)
aeroplane: 98.0%
aeroplane: 100.0%

4.10 <u>Training</u> :

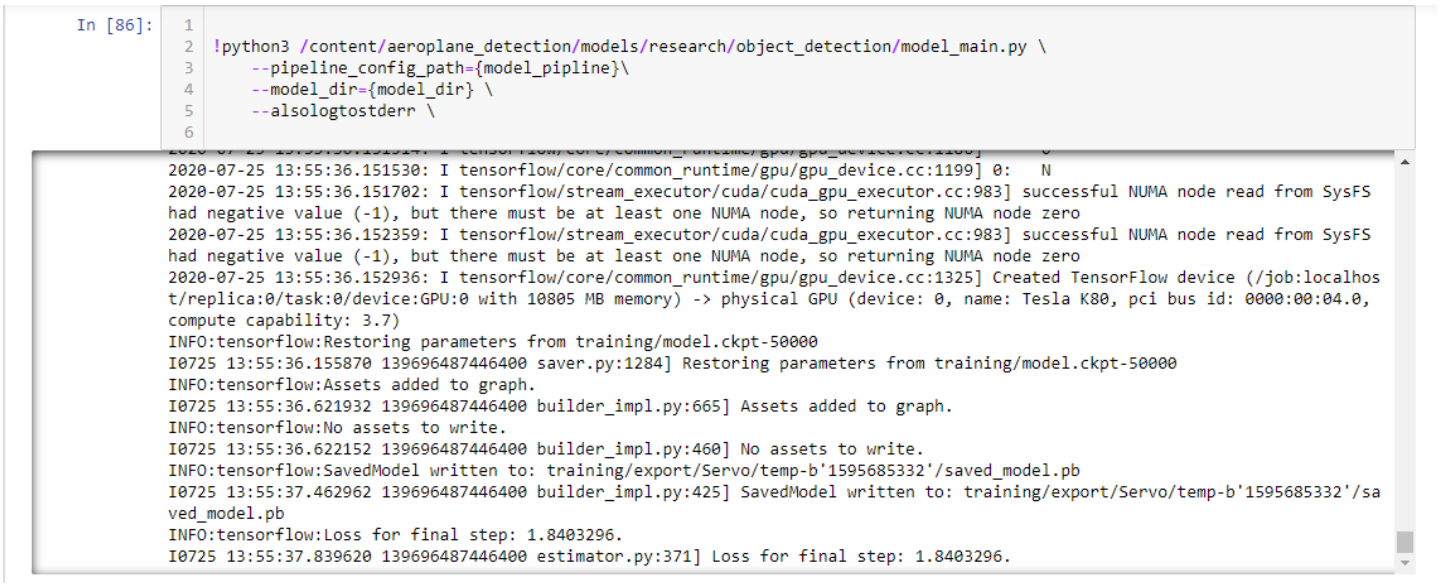
- Here we are training the ssd model which will evaluate and train the model to get the accurate results in the output of detection.
- The below code runs the traning my model
- I have kept the maximum no of steps as 2,00,000 and I have trained my model with 50,000

  Due to the limit of colab ,that is only 12hrs.



```
[ ]  model_dir

     'training/'

[ ]  model_pipline

     '/content/aeroplane_detection/models/research/object_detection/samples/configs/ssd_mobilenet_v2_coco.config'

[ ]
     !python3 /content/aeroplane_detection/models/research/object_detection/model_main.py \
         --pipeline_config_path={model_pipline}\
         --model_dir={model_dir} \
         --alsologtostderr \
```

- The training has 50,000 steps. I have trained my model , it took around 7hrs to train my model and the final loss was 1.8403296 .which is less than 2 which can be accepted..
- For the ssd mode the loss must be less than 2 , it could be better For acquiring the better results



```
In [86]:
!python3 /content/aeroplane_detection/models/research/object_detection/model_main.py \
    --pipeline_config_path={model_pipline}\
    --model_dir={model_dir} \
    --alsologtostderr \
```

```
2020-07-25 13:55:36.151530: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1199] 0:   N
2020-07-25 13:55:36.151702: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:983] successful NUMA node read from SysFS
had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero
2020-07-25 13:55:36.152359: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:983] successful NUMA node read from SysFS
had negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero
2020-07-25 13:55:36.152936: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1325] Created TensorFlow device (/job:localhos
t/replica:0/task:0/device:GPU:0 with 10805 MB memory) -> physical GPU (device: 0, name: Tesla K80, pci bus id: 0000:00:04.0,
compute capability: 3.7)
INFO:tensorflow:Restoring parameters from training/model.ckpt-50000
I0725 13:55:36.155870 139696487446400 saver.py:1284] Restoring parameters from training/model.ckpt-50000
INFO:tensorflow:Assets added to graph.
I0725 13:55:36.621932 139696487446400 builder_impl.py:665] Assets added to graph.
INFO:tensorflow:No assets to write.
I0725 13:55:36.622152 139696487446400 builder_impl.py:460] No assets to write.
INFO:tensorflow:SavedModel written to: training/export/Servo/temp-b'1595685332'/saved_model.pb
I0725 13:55:37.462962 139696487446400 builder_impl.py:425] SavedModel written to: training/export/Servo/temp-b'1595685332'/sa
ved_model.pb
INFO:tensorflow:Loss for final step: 1.8403296.
I0725 13:55:37.839620 139696487446400 estimator.py:371] Loss for final step: 1.8403296.
```

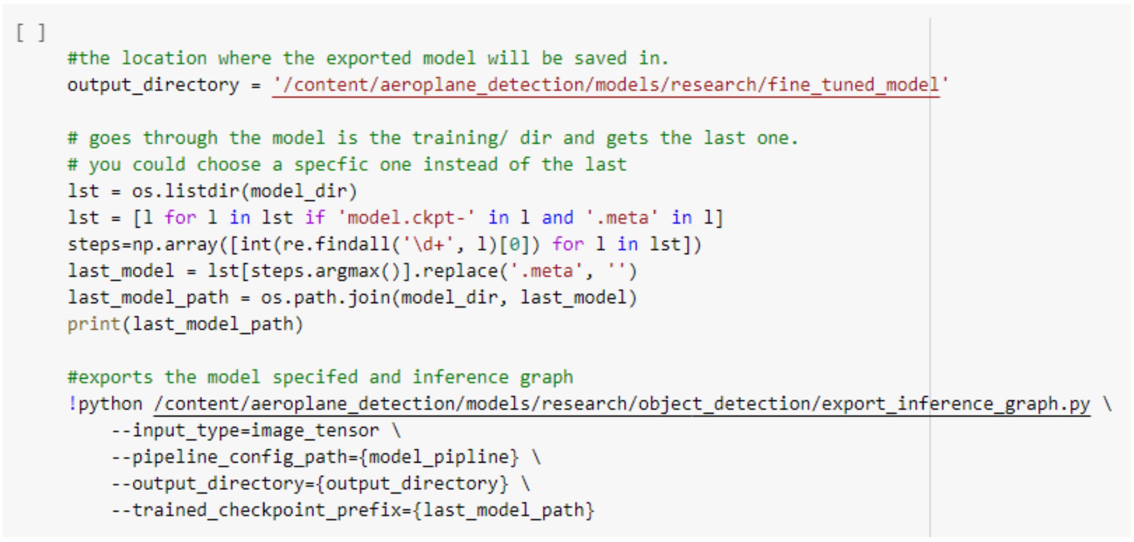
Notes:

1. If the kernel dies, the training will resume from the last checkpoint. Unless you didn't save the `training/` directory somewhere, ex: GDrive. And if the run time disconnects the training does not takes place any more.

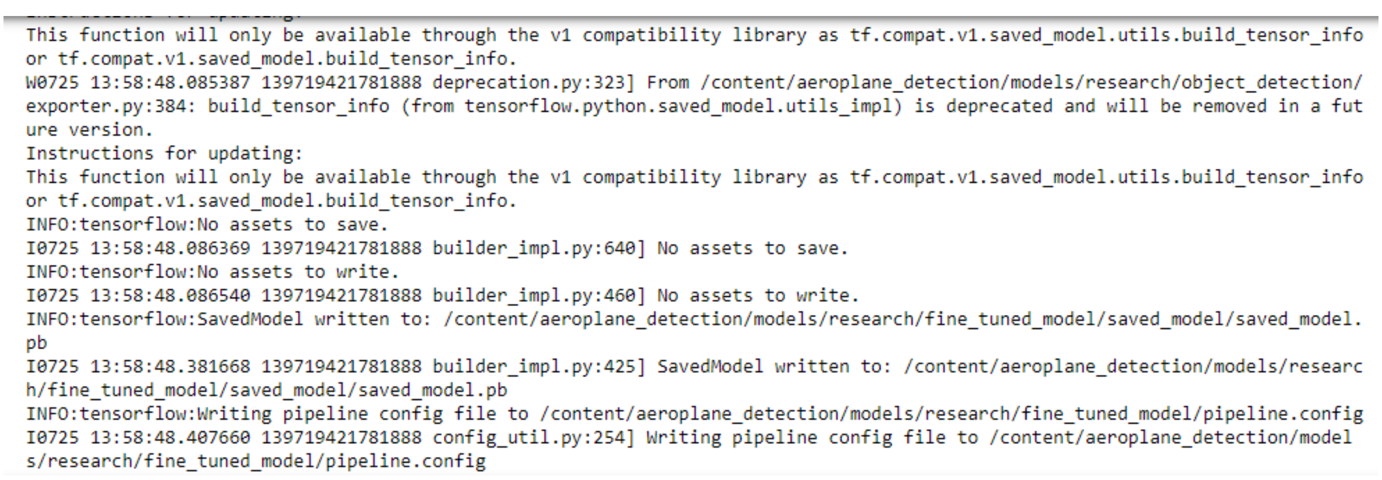
2. If you are changing the above paths, make sure there is no space between the equal sign `=` and the path.

4.11 exporting the model :

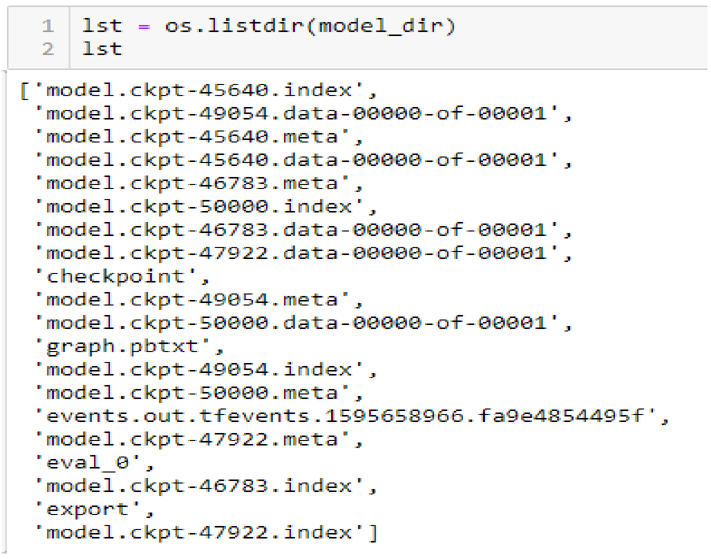
- The below codes will excute my model and test my model.



```
[ ]
    #the location where the exported model will be saved in.
    output_directory = '/content/aeroplane_detection/models/research/fine_tuned_model'

    # goes through the model is the training/ dir and gets the last one.
    # you could choose a specfic one instead of the last
    lst = os.listdir(model_dir)
    lst = [l for l in lst if 'model.ckpt-' in l and '.meta' in l]
    steps=np.array([int(re.findall('\d+', l)[0]) for l in lst])
    last_model = lst[steps.argmax()].replace('.meta', '')
    last_model_path = os.path.join(model_dir, last_model)
    print(last_model_path)

    #exports the model specifed and inference graph
    !python /content/aeroplane_detection/models/research/object_detection/export_inference_graph.py \
        --input_type=image_tensor \
        --pipeline_config_path={model_pipline} \
        --output_directory={output_directory} \
        --trained_checkpoint_prefix={last_model_path}
```



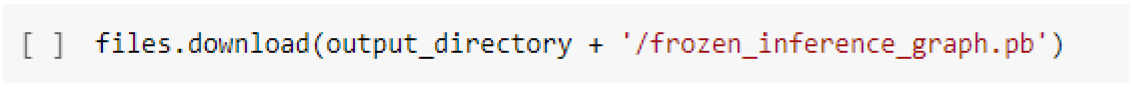
```
This function will only be available through the v1 compatibility library as tf.compat.v1.saved_model.utils.build_tensor_info
or tf.compat.v1.saved_model.build_tensor_info.
W0725 13:58:48.085387 139719421781888 deprecation.py:323] From /content/aeroplane_detection/models/research/object_detection/
exporter.py:384: build_tensor_info (from tensorflow.python.saved_model.utils_impl) is deprecated and will be removed in a fut
ure version.
Instructions for updating:
This function will only be available through the v1 compatibility library as tf.compat.v1.saved_model.utils.build_tensor_info
or tf.compat.v1.saved_model.build_tensor_info.
INFO:tensorflow:No assets to save.
I0725 13:58:48.086369 139719421781888 builder_impl.py:640] No assets to save.
INFO:tensorflow:No assets to write.
I0725 13:58:48.086540 139719421781888 builder_impl.py:460] No assets to write.
INFO:tensorflow:SavedModel written to: /content/aeroplane_detection/models/research/fine_tuned_model/saved_model/saved_model.
pb
I0725 13:58:48.381668 139719421781888 builder_impl.py:425] SavedModel written to: /content/aeroplane_detection/models/researc
h/fine_tuned_model/saved_model/saved_model.pb
INFO:tensorflow:Writing pipeline config file to /content/aeroplane_detection/models/research/fine_tuned_model/pipeline.config
I0725 13:58:48.407660 139719421781888 config_util.py:254] Writing pipeline config file to /content/aeroplane_detection/model
s/research/fine_tuned_model/pipeline.config
```



```
lst = os.listdir(model_dir)
lst
```

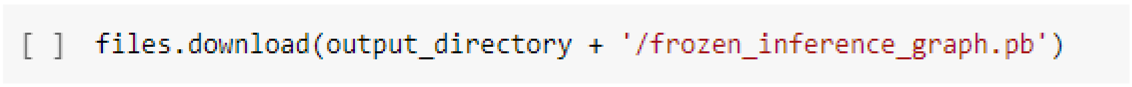
```
['model.ckpt-45640.index',
 'model.ckpt-49054.data-00000-of-00001',
 'model.ckpt-45640.meta',
 'model.ckpt-45640.data-00000-of-00001',
 'model.ckpt-46783.meta',
 'model.ckpt-50000.index',
 'model.ckpt-46783.data-00000-of-00001',
 'model.ckpt-47922.data-00000-of-00001',
 'checkpoint',
 'model.ckpt-49054.meta',
 'model.ckpt-50000.data-00000-of-00001',
 'graph.pbtxt',
 'model.ckpt-49054.index',
 'model.ckpt-50000.meta',
 'events.out.tfevents.1595658966.fa9e4854495f',
 'model.ckpt-47922.meta',
 'eval_0',
 'model.ckpt-46783.index',
 'export',
 'model.ckpt-47922.index']
```

- We need to download the frozen_inference_graph.pd from the fine tuned directory .



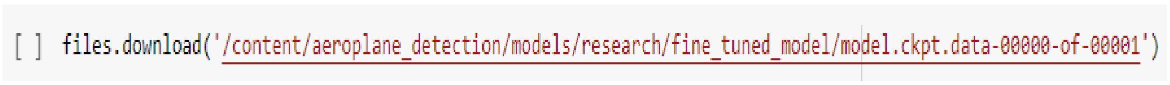
```
[ ]  files.download(output_directory + '/frozen_inference_graph.pb')
```

- Then we need to download the label_map also .



```
[ ]  files.download(output_directory + '/frozen_inference_graph.pb')
```

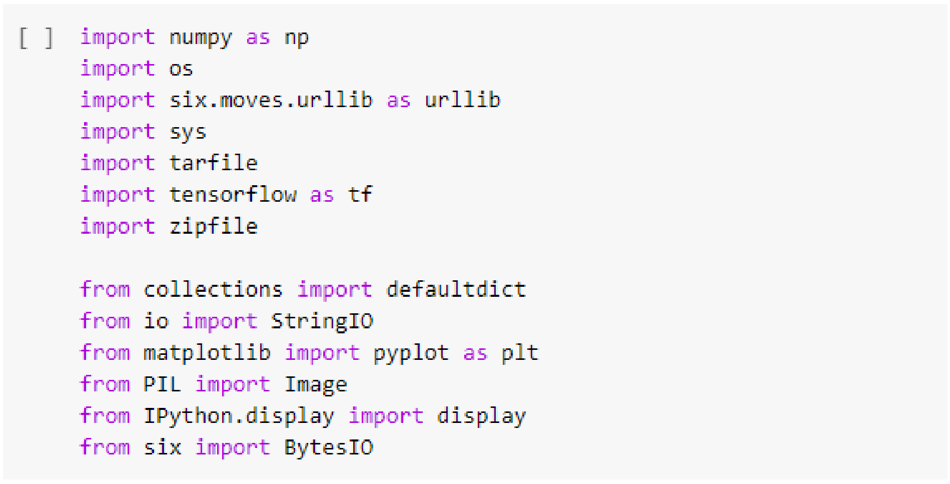
- Again we get another file to be downloaded



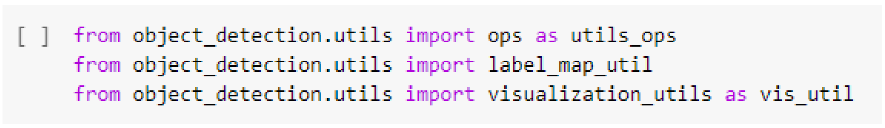
```
[ ] files.download('/content/aeroplane_detection/models/research/fine_tuned_model/model.ckpt.data-00000-of-00001')
```

4.11 Testing the model :

4.11.1 Importing the libraries:

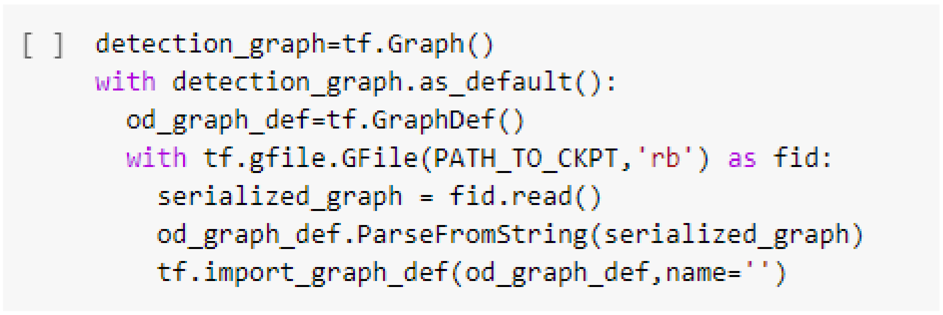


```
[ ]  import numpy as np
     import os
     import six.moves.urllib as urllib
     import sys
     import tarfile
     import tensorflow as tf
     import zipfile

     from collections import defaultdict
     from io import StringIO
     from matplotlib import pyplot as plt
     from PIL import Image
     from IPython.display import display
     from six import BytesIO
```



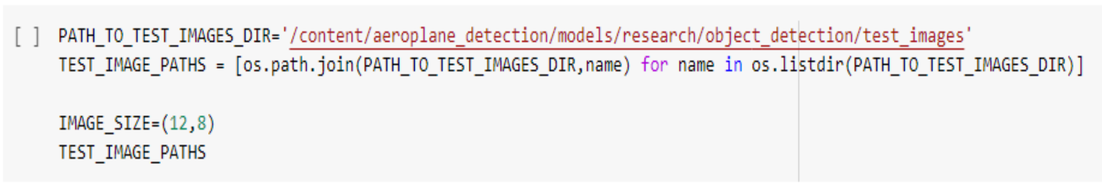
```
[ ]  from object_detection.utils import ops as utils_ops
     from object_detection.utils import label_map_util
     from object_detection.utils import visualization_utils as vis_util
```

- We need to allot the ckpt path to store the frozen graph.

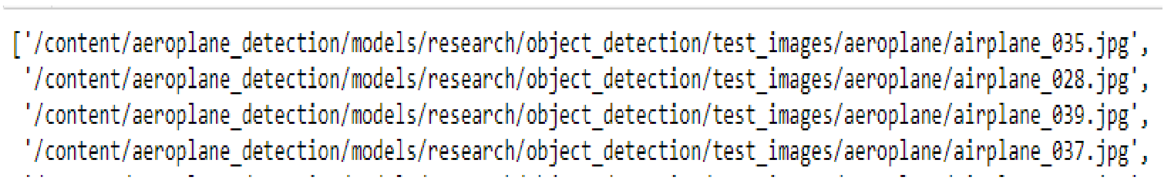


```
[ ]  detection_graph=tf.Graph()
     with detection_graph.as_default():
       od_graph_def=tf.GraphDef()
       with tf.gfile.GFile(PATH_TO_CKPT,'rb') as fid:
         serialized_graph = fid.read()
         od_graph_def.ParseFromString(serialized_graph)
         tf.import_graph_def(od_graph_def,name='')
```

- Now we need create a new folder under the test images , under the newly created folder we need to upload the pics .

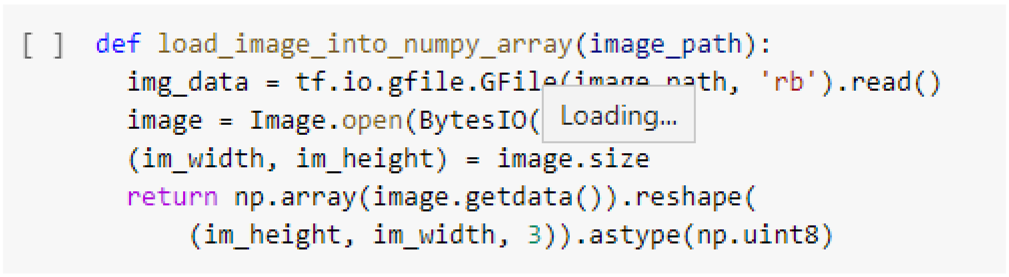


```
[ ]  PATH_TO_TEST_IMAGES_DIR='/content/aeroplane_detection/models/research/object_detection/test_images'
     TEST_IMAGE_PATHS = [os.path.join(PATH_TO_TEST_IMAGES_DIR,name) for name in os.listdir(PATH_TO_TEST_IMAGES_DIR)]

     IMAGE_SIZE=(12,8)
     TEST_IMAGE_PATHS
```



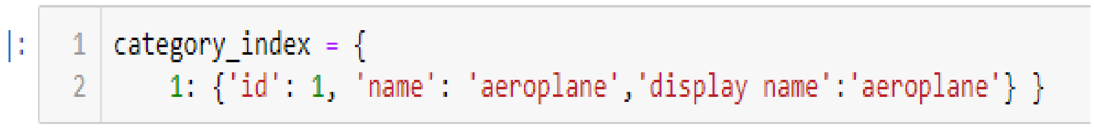
```
['/content/aeroplane_detection/models/research/object_detection/test_images/aeroplane/airplane_035.jpg',
 '/content/aeroplane_detection/models/research/object_detection/test_images/aeroplane/airplane_028.jpg',
 '/content/aeroplane_detection/models/research/object_detection/test_images/aeroplane/airplane_039.jpg',
 '/content/aeroplane_detection/models/research/object_detection/test_images/aeroplane/airplane_037.jpg',
```

- Now we have to define the function to load the images size and to store it into numpy array.

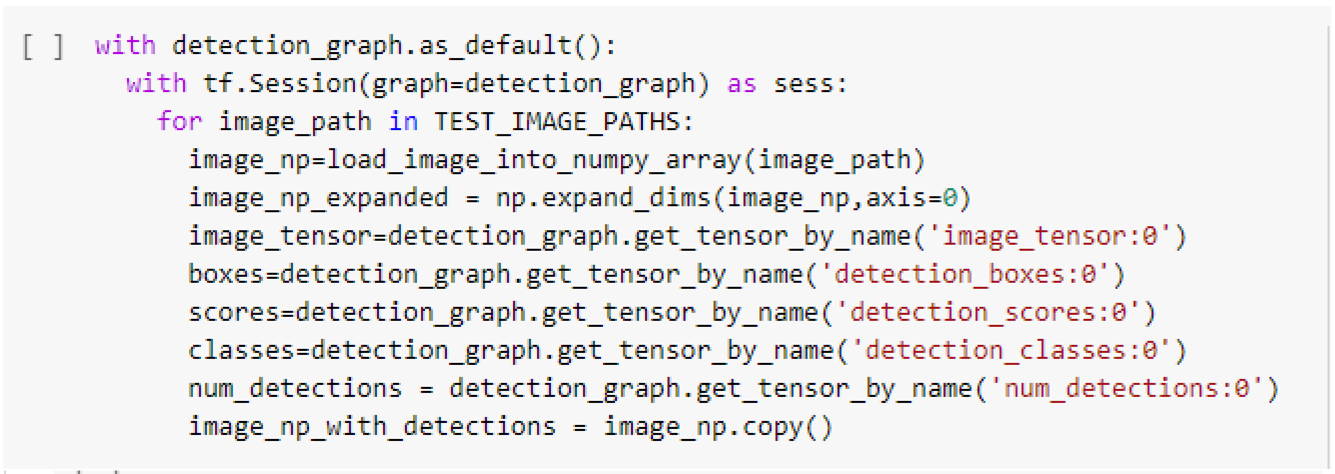


```
[ ]  def load_image_into_numpy_array(image_path):
         img_data = tf.io.gfile.GFile(image_path, 'rb').read()
         image = Image.open(BytesIO( Loading...
         (im_width, im_height) = image.size
         return np.array(image.getdata()).reshape(
             (im_height, im_width, 3)).astype(np.uint8)
```

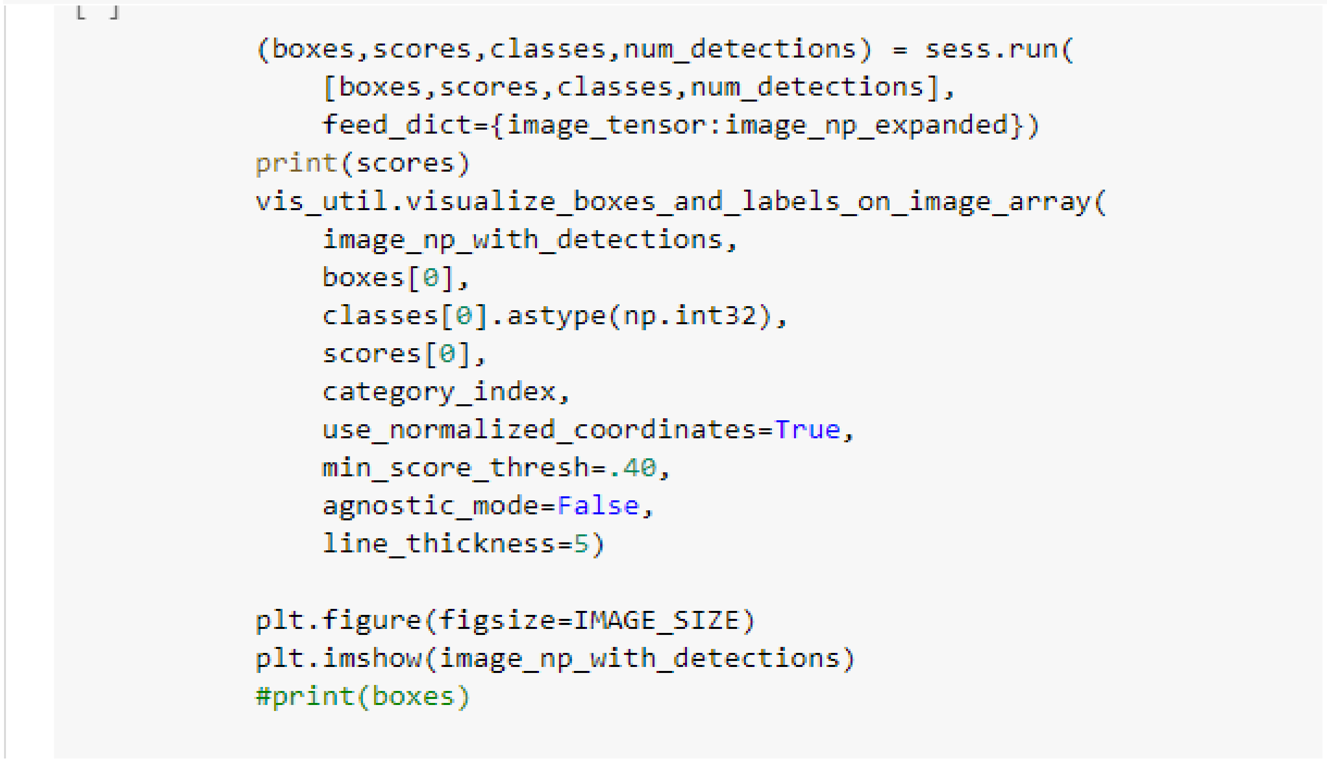
- Categorical index for the diplay of the detection box.



```
category_index = {
    1: {'id': 1, 'name': 'aeroplane','display name':'aeroplane'} }
```



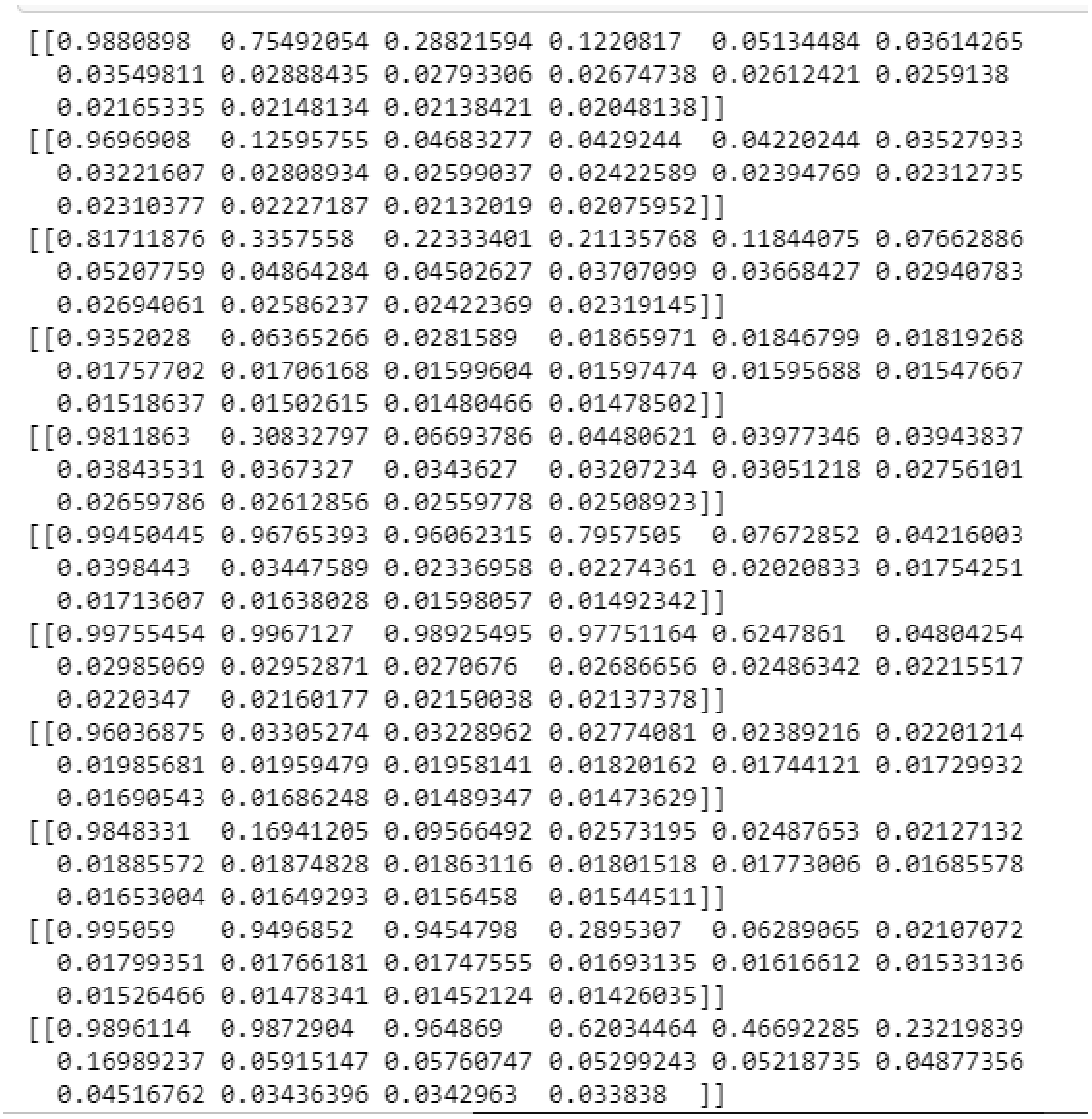
```
[ ] with detection_graph.as_default():
      with tf.Session(graph=detection_graph) as sess:
        for image_path in TEST_IMAGE_PATHS:
          image_np=load_image_into_numpy_array(image_path)
          image_np_expanded = np.expand_dims(image_np,axis=0)
          image_tensor=detection_graph.get_tensor_by_name('image_tensor:0')
          boxes=detection_graph.get_tensor_by_name('detection_boxes:0')
          scores=detection_graph.get_tensor_by_name('detection_scores:0')
          classes=detection_graph.get_tensor_by_name('detection_classes:0')
          num_detections = detection_graph.get_tensor_by_name('num_detections:0')
          image_np_with_detections = image_np.copy()
```



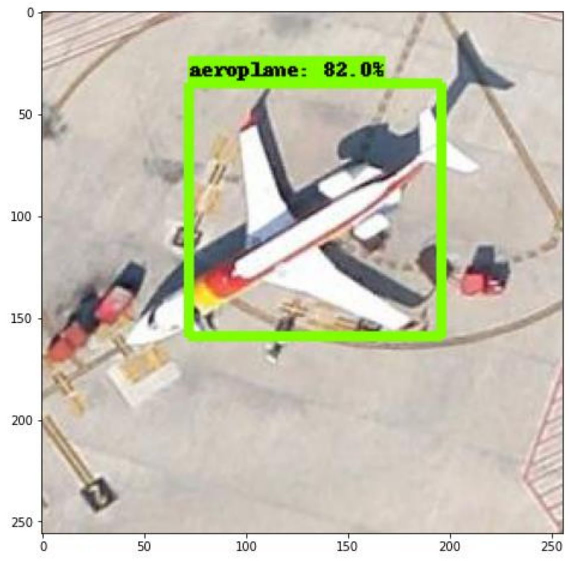
```
          (boxes,scores,classes,num_detections) = sess.run(
              [boxes,scores,classes,num_detections],
              feed_dict={image_tensor:image_np_expanded})
          print(scores)
          vis_util.visualize_boxes_and_labels_on_image_array(
              image_np_with_detections,
              boxes[0],
              classes[0].astype(np.int32),
              scores[0],
              category_index,
              use_normalized_coordinates=True,
              min_score_thresh=.40,
              agnostic_mode=False,
              line_thickness=5)

          plt.figure(figsize=IMAGE_SIZE)
          plt.imshow(image_np_with_detections)
          #print(boxes)
```

4.12 Output of the project :

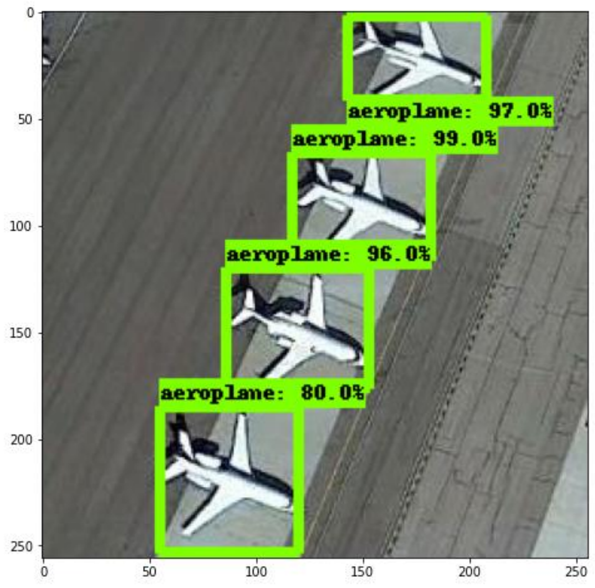
- From the testing of the images it predicts of the boxes on the images.
- The images display the prediction of the boxes and display of the name and how much my Aeroplane is predicted.



```
[[0.9880898  0.75492054 0.28821594 0.1220817  0.05134484 0.03614265
  0.03549811 0.02888435 0.02793306 0.02674738 0.02612421 0.0259138
  0.02165335 0.02148134 0.02138421 0.02048138]]
[[0.9696908  0.12595755 0.04683277 0.0429244  0.04220244 0.03527933
  0.03221607 0.02808934 0.02599037 0.02422589 0.02394769 0.02312735
  0.02310377 0.02227187 0.02132019 0.02075952]]
[[0.81711876 0.3357558  0.22333401 0.21135768 0.11844075 0.07662886
  0.05207759 0.04864284 0.04502627 0.03707099 0.03668427 0.02940783
  0.02694061 0.02586237 0.02422369 0.02319145]]
[[0.9352028  0.06365266 0.0281589  0.01865971 0.01846799 0.01819268
  0.01757702 0.01706168 0.01599604 0.01597474 0.01595688 0.01547667
  0.01518637 0.01502615 0.01480466 0.01478502]]
[[0.9811863  0.30832797 0.06693786 0.04480621 0.03977346 0.03943837
  0.03843531 0.0367327  0.0343627  0.03207234 0.03051218 0.02756101
  0.02659786 0.02612856 0.02559778 0.02508923]]
[[0.99450445 0.96765393 0.96062315 0.7957505  0.07672852 0.04216003
  0.0398443  0.03447589 0.02336958 0.02274361 0.02020833 0.01754251
  0.01713607 0.01638028 0.01598057 0.01492342]]
[[0.99755454 0.9967127  0.98925495 0.97751164 0.6247861  0.04804254
  0.02985069 0.02952871 0.0270676  0.02686656 0.02486342 0.02215517
  0.0220347  0.02160177 0.02150038 0.02137378]]
[[0.96036875 0.03305274 0.03228962 0.02774081 0.02389216 0.02201214
  0.01985681 0.01959479 0.01958141 0.01820162 0.01744121 0.01729932
  0.01690543 0.01686248 0.01489347 0.01473629]]
[[0.9848331  0.16941205 0.09566492 0.02573195 0.02487653 0.02127132
  0.01885572 0.01874828 0.01863116 0.01801518 0.01773006 0.01685578
  0.01653004 0.01649293 0.0156458  0.01544511]]
[[0.995059   0.9496852  0.9454798  0.2895307  0.06289065 0.02107072
  0.01799351 0.01766181 0.01747555 0.01693135 0.01616612 0.01533136
  0.01526466 0.01478341 0.01452124 0.01426035]]
[[0.9896114  0.9872904  0.964869   0.62034464 0.46692285 0.23219839
  0.16989237 0.05915147 0.05760747 0.05299243 0.05218735 0.04877356
  0.04516762 0.03436396 0.0342963  0.033838  ]]
```



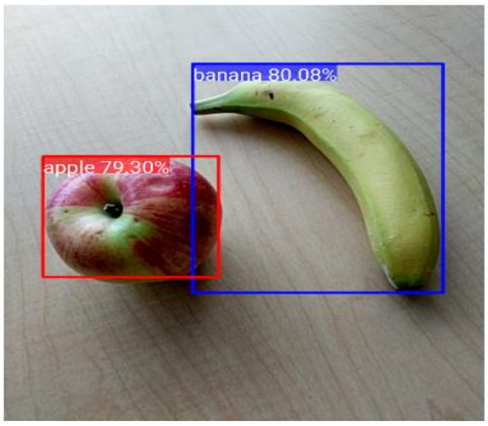
- The above image is predicting my aeroplane accuracy and detecting the aeroplane with the name diplaying with boundary box.

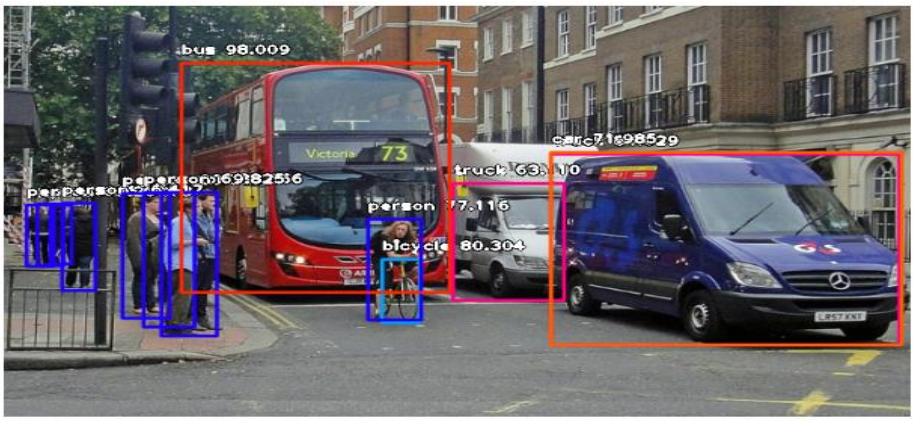


- The above image with several aeroplane detecting the image with accuracy(percentage).

5 Case study :

- The object detection have so many algorithms to predict the output .
- The object detection is done through the deep learning , Artificial Neural Network.
- Examples of the object detection. Here its predicting the objects apple and banana.



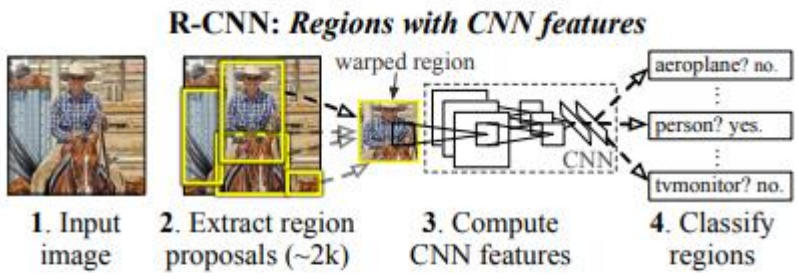


- They are different models , RCNN, FASTER RCNN, SSD , and YOLO.

5.1 R CNN(regional based convolutional network) :

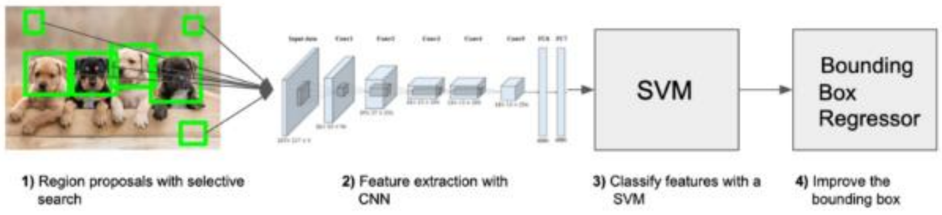
- This technique combines two main approaches: applying high-capacity convolutional neural networks to bottom-up region proposals so as to localize and segment objects; and supervised pre-training for auxiliary tasks.
- . This is followed by domain-specific fine-tuning that yields a high-performance boost. The authors of this paper named the algorithm

R-CNN (Regions with CNN features) because it combines regional proposals with convolutional neural networks.



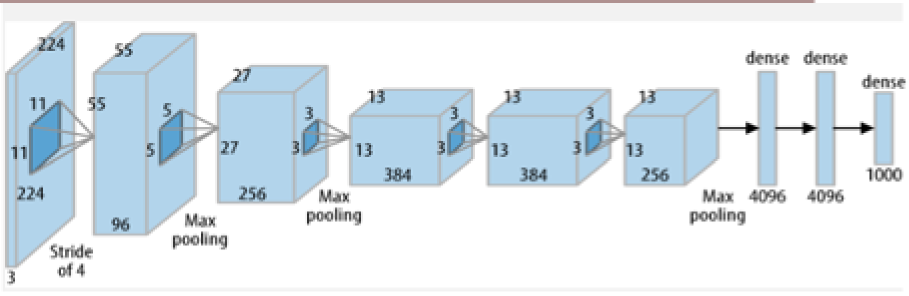
Source

○


SVM
Bounding Box Regressor
1) Region proposals with selective search
2) Feature extraction with CNN
3) Classify features with a SVM
4) Improve the bounding box

5.2 **CNN(**Convulutional Neural Network**) :**

- CNNs are regularized versions of multilayer perceptrons.
- Multilayer perceptrons usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer.
- Convolutional networks were inspired by biological processes.
- In that the connectivity pattern between neurons resembles the organization of the animal visual cortex.



5.3 Faster RCNN(Fast Region-based Convolutional Network):

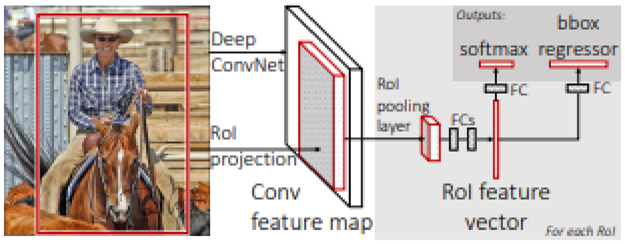
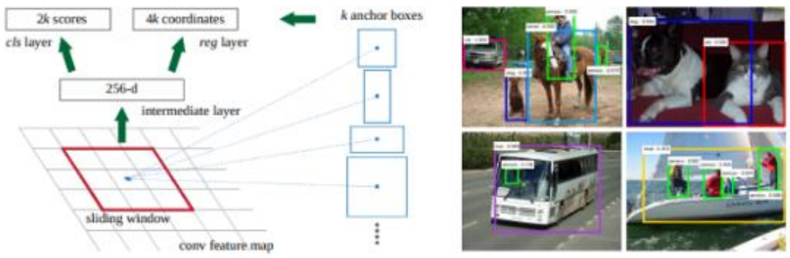


Figure 1. Fast R-CNN architecture. An input image and multiple regions of interest (RoIs) are input into a fully convolutional network. Each RoI is pooled into a fixed-size feature map and then mapped to a feature vector by fully connected layers (FCs). The network has two output vectors per RoI: softmax probabilities and per-class bounding-box regression offsets. The architecture is trained end-to-end with a multi-task loss.

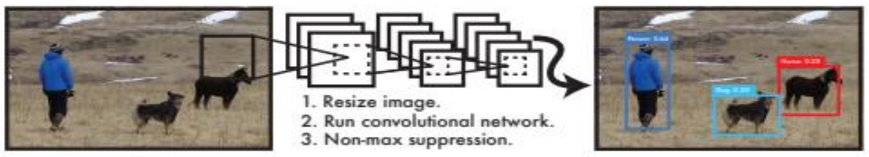
- In comparison to the R-CNN, Fast R-CNN has a higher mean average precision, single stage training, training that updates all network layers, and disk storage isn't required for feature caching.



- In this model, objects are classified and localized using a bounding box and semantic segmentation that classifies each pixel into a set of categories. This model extends Faster R-CNN by adding the prediction of segmentation masks on each Region of Interest. The Mask R-CNN produces two outputs; a class label and a bounding box.

5.4 **YOLO** (you only live once):

- The YOLO models process 45 frames per second in real-time. YOLO views image detection as a regression problem, which makes its pipeline quite simple. It's extremely fast because of this simple pipeline.

- 

- In YOLO, each bounding box is predicted by features from the entire image. Each bounding box has 5 predictions; x, y, w, h, and confidence. (x, y) represents the center of the bounding box relative to the bounds of the grid cell. *w* and *h* are the predicted width and height of the whole image.
- This model is implemented as a convolutional neural network and evaluated on the PASCAL VOC detection dataset. The convolutional layers of the network are responsible for extracting the features, while the fully connected layers predict the coordinates and output probabilities.

CONCLUSION :

- My model has worked properly for detecting the aeroplane .
- This model is very helpful full for the aeroplane management for the detection of aeroplane when the plane is not visible or to locate it under bigger issues.
- For the aeroplane management its is easy detect its older aeroplanes.

REFERENCE

- https://www.geeksforgeeks.org/object-detection-vs-object-recognition-vs-image-segmentation/
- https://towardsdatascience.com/detailed-tutorial-build-your-custom-real-time-object-detector-5ade1017fd2d#5f34
- https://stackabuse.com/object-detection-with-imageai-in-python/
- https://www.tensorflow.org/lite/models/object_detection/overview

github link:
https://github.com/Motupallysundaracharya38/OBJECTDETECTION/blob/master/objectdetect.ipynb

---END--