Car and Pedestrian Detection

A MAJOR PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIRMENT FOR THE AWARD OF THREE-YEAR DIPLOMA

IN

INFORMATION TECHNOLOGY ENABLED SERVICE
&
MANAGMENT

AMBEDKAR INSTITUTE OF TECHNOLOGY

SHAKARPUR, DELHI- 110092

SUBMITTED TO:-

DR. DIPANKAR MISHRA

INCHARGE OF MAJOR PROJECT

(H.O.D ITES&M)

SUBMITTED BY:-

SACHIN KUMAR

(1602111050)

CONTENT

MACHINE LEARNING(M.L)

ACKNOWLEDGEMENT

This project would not have taken shape without the guidance provided by DR. DIPANKAR MISHRA, who helped in the modules of my project and resolved all the technical as well as problems related to project and, for always providing me a helping hand whenever I faced any bottlenecks, in spite of being quite busy with their hectic schedules.

I am thankful to my project supervisor DR. DIPANKAR MISHRA, who gave me this opportunity and provided me all the academic and conceptual support for my project.

Above all I wish to express my heartfelt gratitude to DR. DIPANKAR MISHRA, H.O.D, I.T.E.S&M DEPARTMENT whose support has greatly boosted my self-confidence and will go long way and helping me to reach further milestones and greater heights.

ABSTRACT

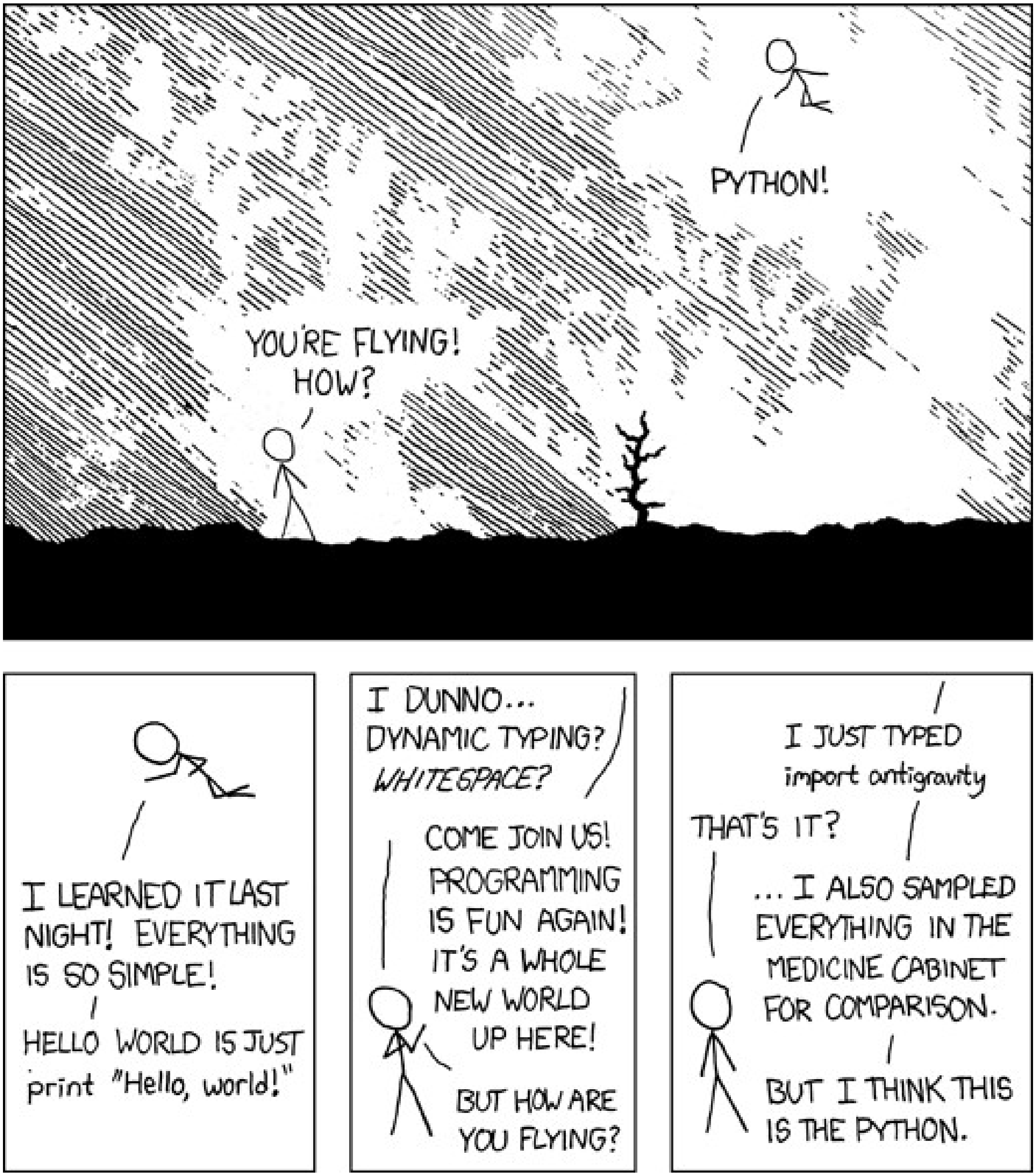
This project is about the **CAR AND PEDESTRIAN DETECTION, (by using OPEN SOURCE COMPUTER VISION and OPENCV is a library developed by INTEL.)** of objects in live scenario, or on a captured photo or video.

This project shows that how Car and Pedestrian Detection works on a part of captured video and tracks the car and Pedestrian of some pedestrian that is similar to the real-life application. Car and Pedestrian is majorly used in Robotics, Self Driving Car, etc. where object classification is very necessary for the machines.

INTRODUCTION

What is Python?

Python is a powerful modern computer programming language. It bears some similarities to FORTRAN, one of the earliest programming languages, but it is much more powerful than FORTRAN. Python allows you to use variables without declaring them (i.e., it determines types implicitly), and it relies on indentation as a control structure. You are not forced to define classes in Python (unlike Java) but you are free to do so when convenient. Python was developed by ***Guido van Rossum***, and it is free software. Free as in "free beer," in that you can obtain Python without spending any money. But Python is also free in other important ways, for example you are free to copy it as many times as you like, and free to study the source code, and make changes to it. There is a worldwide movement behind the idea of free software, initiated in 1983 by Richard Stallman.1 This document focuses on learning Python for the purpose of doing mathematical calculations. We assume the reader has some knowledge of basic mathematics, but we try not to assume any previous exposure to computer programming, although some such exposure would certainly be helpful. Python is a good choice for mathematical calculations, since we can write code quickly, test it easily, and its syntax is similar to the way mathematical ideas are expressed in the mathematical literature. By learning Python you will also be learning a major tool used by many web developers.


PYTHON!
YOU'RE FLYING! HOW?
I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

Why Companies Prefer Python?

Python has top the charts in the recent years over other programming languages like C, C++ and Java and is widely used by the programmers. The language has undergone a drastic change since its release 25 years ago as many add-on features are introduced. The Python 1.0 had the module system of Modula-3 and interacted with Amoeba Operating System with varied functioning tools. Python 2.0 introduced in the year 2000 had features of garbage collector and Unicode Support. Python 3.0 introduced in the year 2008 had a constructive design that avoids duplicate modules and constructs. With the added features, now the companies are using Python 3.5.

The software development companies prefer Python language because of its versatile features and fewer programming codes. Nearly 14% of the programmers use it on the operating systems like UNIX, Linux, Windows and Mac OS. The programmers of big companies use Python as it has created a mark for itself in the software development with characteristic features like-

- Interactive
- Interpreted
- Modular
- Dynamic
- Object-oriented
- Portable
- High level
- Extensible in C++ & C

Advantages or Benefits of Python

The Python language has diversified application in the software development companies such as in gaming, web frameworks and applications, language development, prototyping, graphic design applications, etc. This provides the language a higher plethora over other programming languages used in the industry. Some of its advantages are-

- **Extensive Support Libraries**

It provides large standard libraries that include the areas like string operations, Internet, web service tools, operating system interfaces and protocols. Most of the highly used programming tasks are already scripted into it that limits the length of the codes to be written in Python.

- **Integration Feature**

Python integrates the Enterprise Application Integration that makes it easy to develop Web services by invoking COM or COBRA components. It has powerful control capabilities as it calls directly through C, C++ or Java via Jython. Python also processes XML and other markup languages as it can run on all modern operating systems through same byte code.

- **Improved Programmer's Productivity**

The language has extensive support libraries and clean object-oriented designs that increase two to tenfold of programmer's productivity while using the languages like Java, VB, Perl, C, C++ and C#.

- **Productivity**

With its strong process integration features, unit testing framework and enhanced control capabilities contribute towards the increased speed for most applications and productivity of applications. It is a great option for building scalable multi-protocol network applications.

Limitations or Disadvantages of Python

Python has varied advantageous features, and programmers prefer this language to other programming languages because it is easy to learn and code too. However, this language has still not made its place in some computing arenas that includes Enterprise Development Shops. Therefore, this language may not solve some of the enterprise solutions, and limitations include-

- **Difficulty in Using Other Languages**

The Python lovers become so accustomed to its features and its extensive libraries, so they face problem in learning or working on other programming languages. Python experts may see the declaring of cast "values" or variable "types", syntactic requirements of adding curly braces or semi colons as an onerous task.

- **Weak in Mobile Computing**

Python has made its presence on many desktop and server platforms, but it is seen as a weak language for mobile computing. This is the reason very few mobile applications are built in it like Carbonnelle.

- **Gets Slow in Speed**

Python executes with the help of an interpreter instead of the compiler, which causes it to slow down because compilation and execution help it to work normally. On the other hand, it can be seen that it is fast for many web applications too.

- **Run-time Errors**

The Python language is dynamically typed so it has many design restrictions that are reported by some Python developers. It is even seen that it requires more testing time, and the errors show up when the applications are finally run.

- **Underdeveloped Database Access Layers**

As compared to the popular technologies like JDBC and ODBC, the Python's database access layer is found to be bit underdeveloped and primitive. However, it cannot be applied in the enterprises that need smooth interaction of complex legacy data.

Conclusion

Python is a robust programming language and provides an easy usage of the code lines, maintenance can be handled in a great way, and debugging can be done easily too. It has gained importance across the globe as computer giant Google has made it one of its official programming languages.

Features of Python

Easy-to-learn: Popular (scripting/extension) language, clear and easy syntax, no type declarations, automatic memory management, high-level data types and operations, design to read (more English like syntax) and write (shorter code compared to C, C++, and Java) fast.

High-level Language:
High-level language (closer to human) refers to the higher level of concept from machine language (for example assembly languages). Python is an example of a high-level language like C, C++, Perl, and Java with low-level optimization.

1. **Portable:**
   High level languages are portable, which means they are able to run across all major hardware and software platforms with few or no change in source code. Python is portable and can be used on Linux, Windows, Macintosh, Solaris, FreeBSD, OS/2, Amiga, AROS, AS/400 and many more.
2. **Object-Oriented**: Python is a full-featured object-oriented programming language, with features such as classes, inheritance, objects, and overloading.
3. **Python is Interactive** :
   Python has an interactive console where you get a Python prompt (command line) and interact with the interpreter directly to write and test your programs. This is useful for mathematical programming.
4. **Interpreted** : Python programs are interpreted, takes source code as input, and then compiles (to portable byte-code) each statement and executes it immediately. No need to compiling or linking

5. **Extendable** : Python is often referred to as a "glue" language, meaning that it is capable to work in mixed-language environment. The Python interpreter is easily extended and can add a new built-in function or modules written in C/C++/Java code.
6. **Libraries** : Databases, web services, networking, numerical packages, graphical user interfaces, 3D graphics, others.
7. **Supports** :Support from online Python community

Python Interpreter

- In interactive mode, type Python programs and the interpreter displays the result:
- Type python into your terminal's command line
- After a short message, the >>> symbol will appear
- The above symbol signals the start of a Python interpreter's command line.
- Python interpreter evaluates inputs

How stable is Python?

Very stable. New, stable releases have been coming out roughly every 6 to 18 months since 1991, and this seems likely to continue. Currently there are usually around 18 months between major releases.

The latest stable releases can always be found on the Python download page. There are two recommended production-ready versions at this point in time, because at the moment there are two branches of stable releases: 2.x and 3.x. Python 3.x may be less useful than 2.x, since currently there is more third party software available for Python 2 than for Python 3. Python 2 code will generally not run unchanged in Python 3.

History

The name Python was selected from "Monty Python Flying Circus" which was a British sketch comedy series created by the comedy group Monty Python and broadcast by the BBC from 1969 to 1974.

Python was created in the early 1980s by Guido van Rossum at the National Research Institute for Mathematics and Computer Science in Netherlands.

Python was created as a successor of a language called ABC (All Basic Code) and released publicly in1991. Guido remains Python's principal author, although it includes many contributions from active user community.

Between 1991 and 2001 there are several versions released, current stable release is 3.2. In 2001 the Python Software Foundation (PSF) was formed, a non-profit organization created specifically to own Python-related Intellectual Property. Zope Corporation is a sponsoring member of the PSF.

Major uses of Python

- System utilities (system admin tools, command line programs).

- Web Development.
- Graphical User Interfaces (Tkinter, gtk, Qt).
- Internet scripting.
- Embedded scripting.
- Database access and programming.
- Game programming.
- Rapid prototyping and development.
- Distributed programming

Organizations Using Python (sector wise)

- **Web Development :** Yahoo Maps, Yahoo Groups, Google, Zope Corporation, Ultraseek, Linux Weekly News, ElasticHosts Cloud Servers, Mojam.com, hunch, Shopzilla, Movieplayer.it, Multiplayer.it.
- **Games:** Battlefield 2, Crystal Space, Star Trek Bridge Commander, The Temple of Elemental Evil, Vampire: The Masquerade: Bloodlines, Civilization 4, QuArK (Quake Army Knife)
- **Graphics :** Industrial Light & Magic, Walt Disney Feature Animation, HKS, Inc. (ABAQUS/CAE), RoboFog, Caligari Corporation, Blender 3D, Jasc Software, Paint Shop Pro.
- **Financial :** Altis Investment Management, ABN AMRO Bank, Treasury Systems, Bellco Credit Union, Journyx Timesheet and Resource Management Software.

- **Science :** National Weather Service, Radar Remote Sensing Group, Applied Maths, Biosoft, The National Research Council of Canada, Los Alamos National Laboratory (LANL) Theoretical Physics Division, AlphaGene, Inc., LLNL, NASA, Swedish Meteorological and Hydrological Institute (SMHI), Environmental Systems Research Institute (ESRI), Objexx Engineering, Nmag Computational Micromagnetics
- **Electronic Design Automation:** Ciranova, Productivity Design Tools, Object Domain, Pardus, Red Hat, SGI, Inc., MCI Worldcom, Nokia,
- **Education** : University of California, Irvine, Smeal College of Business, The Pennsylvania State University, New Zealand Digital Library, IT Certification Exam preparation, SchoolTool,
- **Business Software :** Raven Bear Systems Corporation, Thawte Consulting, Advanced Management Solutions Inc., IBM, Arakn<E9>, RealNetworks, dSPACE, Escom, The Tiny Company, Nexedi, Piensa Technologies - Bufete Consultor de Mexico, Nektra, WuBook.

Data types in Python

Every value in Python has a datatype. Since everything is an object in Python programming, data types are actually classes and variables are instance (object) of these classes.

There are various data types in Python. Some of the important types are listed below.

- Python Numbers

Integers, floating point numbers and complex numbers falls under Python numbers category. They are defined as int, float and complex class in Python.

We can use the type() function to know which class a variable or a value belongs to and the isinstance() function to check if an object belongs to a particular class.

```
a = 5
print(a, "is of type", type(a))
a = 2.0
print(a, "is of type", type(a))
a = 1+2j
print(a, "is complex number?", isinstance(1+2j,complex))
```

Integers can be of any length, it is only limited by the memory available.

A floating point number is accurate up to 15 decimal places. Integer and floating points are separated by decimal points. 1 is integer, 1.0 is floating point number.

Complex numbers are written in the form, x + yj, where x is the real part and y is the imaginary part. Here are some examples.

```
>>> a = 1234567890123456789
>>> a
1234567890123456789
>>> b = 0.1234567890123456789
>>> b
0.12345678901234568
>>> c = 1+2j
>>> c
(1+2j)
```

Notice that the float variable b got truncated.

- ## Python List

List is an ordered sequence of items. It is one of the most used datatype in Python and is very flexible. All the items in a list do not need to be of the same type.

Declaring a list is pretty straight forward. Items separated by commas are enclosed within brackets [ ].

```
>>> a = [1, 2.2, 'python']
```

We can use the slicing operator [ ] to extract an item or a range of items from a list. Index starts form 0 in Python.

```
a = [5,10,15,20,25,30,35,40]
# a[2] = 15
print("a[2] = ", a[2])
# a[0:3] = [5, 10, 15]
print("a[0:3] = ", a[0:3])
# a[5:] = [30, 35, 40]
print("a[5:] = ", a[5:])
```

Lists are mutable, meaning, value of elements of a list can be altered.

```
>>> a = [1,2,3]
>>> a[2]=4
>>> a
```

```
[1, 2, 4]
```

- ## Python Tuple

Tuple is an ordered sequence of items same as list.The only difference is that tuples are immutable. Tuples once created cannot be modified.

Tuples are used to write-protect data and are usually faster than list as it cannot change dynamically.

It is defined within parentheses () where items are separated by commas.

```
>>> t = (5,'program', 1+3j)
```

We can use the slicing operator [] to extract items but we cannot change its value.

```
0t = (5,'program', 1+3j)
# t[1] = 'program'
print("t[1] = ", t[1])
# t[0:3] = (5, 'program', (1+3j))
print("t[0:3] = ", t[0:3])
# Generates error
# Tuples are immutable
t[0] = 10
```

- ## Python Strings

String is sequence of Unicode characters. We can use single quotes or double quotes to represent strings. Multi-line strings can be denoted using triple quotes, ''' or """.

```
>>> s = "This is a string"
>>> s = '''a multiline
```

Like list and tuple, slicing operator [ ] can be used with string. Strings are immutable.

```
s = 'Hello world!'
# s[4] = 'o'
print("s[4] = ", s[4])
# s[6:11] = 'world'
print("s[6:11] = ", s[6:11])
# Generates error
# Strings are immutable in Python
s[5] ='d'
```

- **Python Set**

Set is an unordered collection of unique items. Set is defined by values separated by comma inside braces { }. Items in a set are not ordered.

```
a = {5,2,3,1,4}
# printing set variable
print("a = ", a)
# data type of variable a
print(type(a))
```

We can perform set operations like union, intersection on two sets. Set have unique values. They eliminate duplicates.

```
>>> a = {1,2,2,3,3,3}
>>> a
{1, 2, 3}
```

Since, set are unordered collection, indexing has no meaning. Hence the slicing operator [] does not work.

```
>>> a = {1,2,3}
>>> a[1]
Traceback (most recent call last):
  File "<string>", line 301, in runcode
  File "<interactive input>", line 1, in <module>
TypeError: 'set' object does not support indexing
```

- **Python Dictionary**

Dictionary is an unordered collection of key-value pairs.

It is generally used when we have a huge amount of data. Dictionaries are optimized for retrieving data. We must know the key to retrieve the value.

In Python, dictionaries are defined within braces {} with each item being a pair in the form key:value. Key and value can be of any type.

```
>>> d = {1:'value','key':2}
>>> type(d)
<class 'dict'>
```

We use key to retrieve the respective value. But not the other way around.

```
d = {1:'value','key':2}
print(type(d))
print("d[1] = ", d[1]);
print("d['key'] = ", d['key']);
# Generates error
print("d[2] = ", d[2]);
```

SOME ADVANCED TOPICS OF PYTHON

1. DATASTRUCTURES

A data structure represents logical arrangement of elements in memory in a particular model, Data structures are also known as abstract data types(ADTs).

Stacks, linked lists and queues are important data structure which are most used in software.

To create stacks, linked and queues are important data structures which are most used in software.

A linked list is a set of nodes such that each node contains a data field to store data and two link fields to refer to the previous node and next node.

Insertion, deletion and replacing the elements are important operation in case of a linked list.

A stack represents a group of elements arranged in memory in LIFO (LAST IN FIRST OUT) manner.Push, pop and peep (or peek) operations are important in case of stacks.

A queue is a data structure where the first element which entered the queue will come out first, this is called FIFO (FIRST IN FIRST OUT) order.

In queue, the elements are added only at the rear (or back) side of the queue and they are removed from the front of the queue. Adding the elements and removing the elements are the two important operations on queues.

A double-ended queue (or deque) is a queue where elements can be inserting or deleted from both ends. Deques are more efficient than the normal queues in terms of memory usage and speed.

Adding elements at the front and at the rear, deleting the elements at the front and rear are the two important operations that one can perform on a deque.

2. DECORATOR

Decorators belong most probably to the most beautiful and most powerful design possibilities in Python, but at the same time the concept is considered by many as complicated to get into. To be precise, the usage of decorates is very easy, but writing decorators can be complicated, especially if you are not experienced with decorators and some functional programming concepts.

Even though it is the same underlying concept, we have two different kinds of decorators in Python:

- Function decorators
- Class decorators

A decorator in Python is any callable Python object that is used to modify a function or a class. A reference to a function "func" or a class "C" is passed to a decorator and the decorator returns a modified function or class. The modified functions or classes usually contain calls to the original function "func" or class "C".

3. REGULAR EXPRESSIONS

Regular expressions (called REs, or regexes, or regex patterns) are essentially a tiny, highly specialized programming language embedded inside Python and made available through the re module. Using this little language, you specify the rules for the set of possible strings that you want to match; this set might contain English sentences, or e-mail addresses, or TeX commands, or anything you like. You can then ask questions such as “Does this string match the pattern?”, or “Is there a match for the pattern anywhere in this string?”. You can also use REs to modify a string or to split it apart in various ways.

Regular expression patterns are compiled into a series of bytecodes which are then executed by a matching engine written in C. For advanced use, it may be necessary to pay careful attention to how the engine will execute a given RE, and write the RE in a certain way in order to produce bytecode that runs faster. Optimization isn’t covered in this document, because it requires that you have a good understanding of the matching engine’s internals.

The regular expression language is relatively small and restricted, so not all possible string processing tasks can be done using regular expressions. There are also tasks that *can* be done with regular expressions, but the expressions turn out to be very complicated. In these cases, you may be better off writing Python code to do the processing; while Python code will be slower than an elaborate regular expression, it will also probably be more understandable.

These are some identifiers used in regular expressions:-

1. #w - read a single char A_Z,a-z,0-9(*Note space and special char is not readable in this)
2. #+ - match 1 char or more
3. #s - read space(for more then one space use + with \s)
4. #. - read single char (but it can read everything)
5. #[]- works as or.
6. #find all - when u have to give more hand one conditions.
7. #\d - read integer only one integer.(add + to read whole integer value)
8. #re.I - called as flag(it ignores the case of the letter)
9. #^ - match at the beginning(used to match any letter at the beginning)
10. #$ - match at the ending(used to match any letter at the end)

4. GUI- GRAPHICAL USER INTERFACE (USING TKINTER)

Python provides various options for developing graphical user interfaces (GUIs). Most important are listed below.

- **Tkinter** − Tkinter is the Python interface to the Tk GUI toolkit shipped with Python. We would look this option in this chapter.
- **wxPython** − This is an open-source Python interface for wxWindows .
- **JPython** − JPython is a Python port for Java which gives Python scripts seamless access to Java class libraries on the local machine.

Tkinter Programming

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

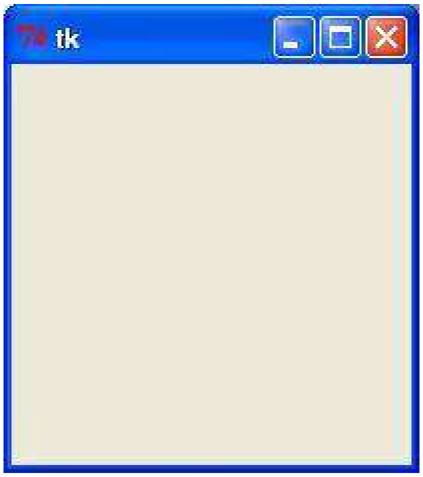
Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps −

- Import the *Tkinter* module.
- Create the GUI application main window.
- Add one or more of the above-mentioned widgets to the GUI application.
- Enter the main event loop to take action against each event triggered by the user.

Example

```
import Tkinter

top = Tkinter.Tk()

# Code to add widgets will go here...

top.mainloop()
```

This would create a following window –



5. ***<u>NETWORK PROGRAMMING INTRODCTION:</u>***

<u>SOCKET PROGRAMMING</u>

Python provides two levels of access to network services. At a low level, you can access the basic socket support in the underlying operating system, which allows you to implement clients and servers for both connection-oriented and connectionless protocols.

Python also has libraries that provide higher-level access to specific application-level network protocols, such as FTP, HTTP, and so on.

What is Sockets?

Sockets are the endpoints of a bidirectional communications channel. Sockets may communicate within a process, between processes on the same machine, or between processes on different continents.

Sockets may be implemented over a number of different channel types: Unix domain sockets, TCP, UDP, and so on. The socket library provides specific classes for handling the common transports as well as a generic interface for handling the rest.

6. ***MY SQL IN PYTHON (DATABASE ACCESS INTRODUCTION)***

The Python standard for database interfaces is the Python DB-API. Most Python database interfaces adhere to this standard.

You can choose the right database for your application. Python Database API supports a wide range of database servers such as −

- GadFly
- mSQL
- MySQL
- PostgreSQL
- Microsoft SQL Server 2000
- Informix
- Interbase
- Oracle
- Sybase

Here is the list of available Python database interfaces: Python Database Interfaces and APIs. You must download a separate DB API module for each database you need to access. For example, if you need to access an Oracle database as well as a MySQL database, you must download both the Oracle and the MySQL database modules.

The DB API provides a minimal standard for working with databases using Python structures and syntax wherever possible. This API includes the following −

Importing the API module.

Acquiring a connection with the database.

Issuing SQL statements and stored procedures.

Closing the connection

We would learn all the concepts using MySQL, so let us talk about MySQLdb module.

What is MySQLdb?

MySQLdb is an interface for connecting to a MySQL database server from Python. It implements the Python Database API v2.0 and is built on top of the MySQL C API.

SAMPLE PROGRAM FOR SQL DATABASE(CREATING DATABASE):-

```
import MySQLdb

# Open database connection
```

```
db = MySQLdb.connect("localhost","testuser","test123","TESTDB" )

# prepare a cursor object using cursor() method
cursor = db.cursor()

# Drop table if it already exist using execute() method.
cursor.execute("DROP TABLE IF EXISTS EMPLOYEE")

# Create table as per requirement
sql = """CREATE TABLE EMPLOYEE (
         FIRST_NAME  CHAR(20) NOT NULL,
         LAST_NAME  CHAR(20),
         AGE INT,
         SEX CHAR(1),
         INCOME FLOAT )"""

cursor.execute(sql)

# disconnect from server
db.close()
```

MACHINE LEARNING(M.L)

Machine learning (ML) is the scientific study of algorithms and statistical models that computer systems use to effectively perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence.

Machine learning algorithms build a mathematical model of sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task.

Machine learning algorithms are used in a wide variety of applications, such as email filtering, and computer vision, where it is infeasible to develop an algorithm of specific instructions for performing the task.

Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a field of study within machine learning, and focuses on exploratory data analysis through unsupervised learning.In its application across business problems, machine learning is also referred to as predictive analytics.

Types of learning algorithms

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

Supervised and semi-supervised learning

Supervised learning algorithms build a mathematical model of a set of data that contains both the inputs and the desired outputs. The data is known as training data, and consists of a set of training examples. Each training example has one or more inputs and a desired output, also known as a supervisory signal. In the case of semi-supervised learning algorithms, some of the training examples are missing the desired output. In the mathematical model, each training example is represented by an array or vector, and the training data by a matrix. Through iterative optimization of an objective function, supervised learning algorithms learn a function that can be used to predict the output associated with new inputs. An optimal function will allow the algorithm to correctly determine the output for inputs that were not a part of the training data. An algorithm that improves the accuracy of its outputs or predictions over time is said to have learned to perform that task.

Supervised learning algorithms
include classification and regression.Classification algorithms are used when the outputs are restricted to a limited set of values, and regression algorithms are used when the outputs may have any numerical value within a range. Similarity learning is an area of supervised machine learning closely related to regression and classification, but the goal is to learn from examples using a similarity function that measures how similar or related two objects are. It has applications
in ranking, recommendation systems, visual identity tracking, face verification, and speaker verification.

Unsupervised learning

Unsupervised learning algorithms take a set of data that contains only inputs, and find structure in the data, like grouping or clustering of data points. The algorithms therefore learn from test data that has not been labeled, classified or categorized. Instead of responding to feedback, unsupervised learning algorithms identify commonalities in the data and react based on the presence or absence of such commonalities in each new piece of data. A central application of unsupervised learning is in the field of density estimation in statistics, though unsupervised learning encompasses other domains involving summarizing and explaining data features.

Cluster analysis is the assignment of a set of observations into subsets (called *clusters*) so that observations within the same cluster are similar according to one or more predesignated criteria, while observations drawn from different clusters are dissimilar. Different clustering techniques make different assumptions on the structure of the data, often defined by some *similarity metric* and evaluated, for example, by *internal compactness*, or the similarity between members of the same cluster, and *separation*, the difference between clusters. Other methods are based on *estimated density* and *graph connectivity*.

Reinforcement learning

Reinforcement learning is an area of machine learning concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward. Due to its generality, the field is studied in many other disciplines, such as game theory, control theory, operations research, information theory, simulation-based optimization, multi-agent systems, swarm intelligence, statistics and genetic algorithms. In machine learning, the environment is typically represented as a Markov Decision Process (MDP). Many reinforcement learning algorithms use dynamic programming techniques. Reinforcement learning algorithms do not assume knowledge of an exact mathematical model of the MDP, and are used when exact models are infeasible. Reinforcement learning algorithms are used in autonomous vehicles or in learning to play a game against a human opponent.

APPLICATIONS OF M.L

There are many applications for machine learning, including:

- Agriculture
- Anatomy
- Adaptive websites
- Affective computing
- Bioinformatics
- Brain–machine interfaces
- Cheminformatics
- Computer Networks
- Computer vision
- Credit-card fraud detection
- Data quality
- DNA sequence classification
- Economics

- Financial market analysis
- General game playing
- Handwriting recognition
- Information retrieval
- Insurance
- Internet fraud detection
- Linguistics
- Machine learning control
- Machine perception
- Machine translation
- Marketing
- Medical diagnosis
- Natural language processing

SOME IMPORTANT PACKAGES USED IN M.L

NUMPY

What is NumPy?

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data.
Arbitrary data-types can be defined using Numpy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

The ndarray data structure

The core functionality of NumPy is its "ndarray", for *n*-dimensional array, data structure. These arrays are strided views on memory. In contrast to Python's built-in list data structure (which, despite the name, is a dynamic array), these arrays are homogeneously typed: all elements of a single array must be of the same type.

Such arrays can also be views into memory buffers allocated by C/C++, Cython, and Fortran extensions to the CPython interpreter without the need to copy data around, giving a degree of compatibility with existing numerical libraries. This functionality is exploited by the SciPy package, which wraps a number of such libraries (notably BLAS and LAPACK). NumPy has built-in support for memory-mapped ndarrays.

PANDAS

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals.

In 2008, developer Wes McKinney started developing pandas when in need of high performance, flexible tool for analysis of data.

Prior to Pandas, Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data — load, prepare, manipulate, model, and analyze.

Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Key Features of Pandas

- Fast and efficient DataFrame object with default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of date sets.
- Label-based slicing, indexing and subsetting of large data sets.
- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.
- High performance merging and joining of data.
- Time Series functionality.

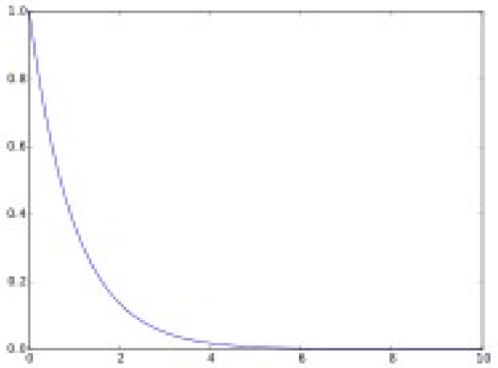
MATPLOTLIB

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged.[3] SciPy makes use of Matplotlib.

EXAMPLES FOR MATPLOTLIB

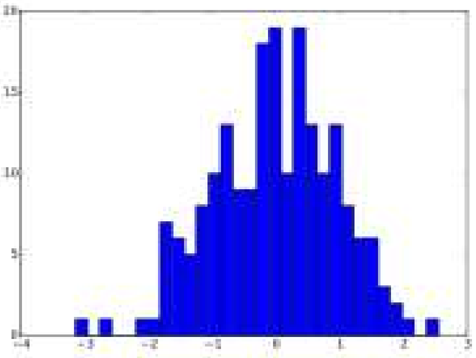
Line plot

```
>>> import matplotlib.pyplot as plt

>>> import numpy as np

>>> a = np.linspace(0, 10, 100)

>>> b = np.exp(-a)

>>> plt.plot(a, b)

>>> plt.show()
```



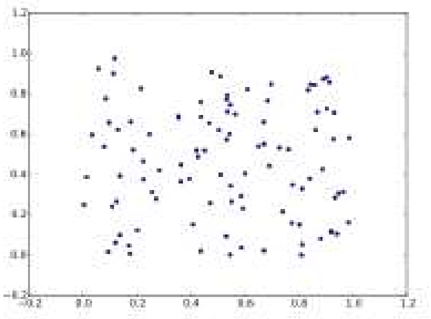
Histogram

```
>>> import matplotlib.pyplot as plt
>>> from numpy.random import normal,rand
>>> x = normal(size=200)
>>> plt.hist(x, bins=30)
>>> plt.show()
```



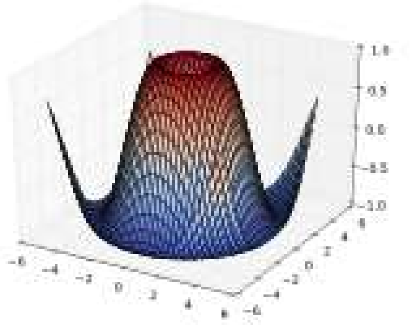
Scatter plot

```
>>> import matplotlib.pyplot as plt
>>> from numpy.random import rand
>>> a = rand(100)
>>> b = rand(100)

>>> plt.scatter(a, b)

>>> plt.show()
```



3D plot

```
>>> from matplotlib import cm

>>> from mpl_toolkits.mplot3d import Axes3D

>>> import matplotlib.pyplot as plt

>>> import numpy as np

>>> fig = plt.figure()

>>> ax = fig.gca(projection='3d')

>>> X = np.arange(-5, 5, 0.25)

>>> Y = np.arange(-5, 5, 0.25)

>>> X, Y = np.meshgrid(X, Y)

>>> R = np.sqrt(X**2 + Y**2)
>>> Z = np.sin(R)

>>> surf = ax.plot_surface(X, Y, Z, rstride=1, cstride=1,
cmap=cm.coolwarm)

>>> plt.show()
```



scikit-learn

Scikit-learn (formerly scikits.learn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

What is scikit-learn?

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python.

It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

The library is built upon the SciPy (Scientific Python) that must be installed before you can use scikit-learn. This stack that includes:

- **NumPy**: Base n-dimensional array package
- **SciPy**: Fundamental library for scientific computing
- **Matplotlib**: Comprehensive 2D/3D plotting
- **IPython**: Enhanced interactive console
- **Sympy**: Symbolic mathematics
- **Pandas**: Data structures and analysis

Extensions or modules for SciPy care conventionally named SciKits. As such, the module provides learning algorithms and is named scikit-learn.

The vision for the library is a level of robustness and support required for use in production systems. This means a deep focus on concerns such as ease of use, code quality, collaboration, documentation and performance.

Although the interface is Python, c-libraries are leverage for performance such as numpy for arrays and matrix operations, LAPACK, LibSVM and the careful use of cython.

REGRESSON

Introduction

Linear and Logistic regressions are usually the first algorithms people learn in data science. Due to their popularity, a lot of analysts even end up thinking that they are the only form of regressions. The ones who are slightly more involved think that they are the most important amongst all forms of regression analysis.

The truth is that there are innumerable forms of regressions, which can be performed. Each form has its own importance and a specific condition where they are best suited to apply. In this article, I have explained the most commonly used 7 forms of regressions in data science in a simple manner. Through this article, I also hope that people develop an idea of the breadth of regressions, instead of just applying linear / logistic regression to every problem they come across and hoping that they would just fit!

What is Regression Analysis?

Regression analysis is a form of predictive modelling technique which investigates the relationship between
a dependent (target) and independent variable (s) (predictor). This technique is used for forecasting, time series modelling and finding the causal effect relationship between the variables. For example, relationship between rash driving and number of road accidents by a driver is best studied through regression.

Regression analysis is an important tool for modelling and analyzing data. Here, we fit a curve / line to the data points, in such a manner that the differences between the distances of data points from the curve or line is minimized. I'll explain this in more details in coming sections.

Why do we use Regression Analysis?

As mentioned above, regression analysis estimates the relationship between two or more variables. Let's understand this with an easy example:

Let's say, you want to estimate growth in sales of a company based on current economic conditions. You have the recent company data which indicates that the growth in sales is around two and a half times the growth in the economy. Using this insight, we can predict future sales of the company based on current & past information.

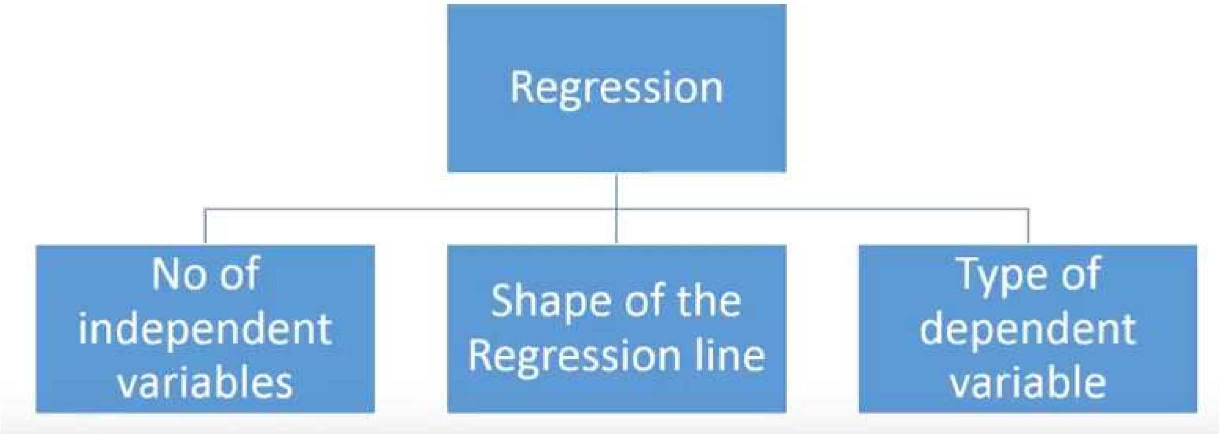
There are multiple benefits of using regression analysis. They are as follows:

1. It indicates the significant relationships between dependent variable and independent variable.
2. It indicates the strength of impact of multiple independent variables on a dependent variable.

Regression analysis also allows us to compare the effects of variables measured on different scales, such as the effect of price changes and the number of promotional activities. These benefits help market researchers / data analysts / data scientists to eliminate and evaluate the best set of variables to be used for building predictive models.

How many types of regression techniques do we have?

There are various kinds of regression techniques available to make predictions. These techniques are mostly driven by three metrics (number of independent variables, type of dependent variables and shape of regression line). We'll discuss them in detail in the following sections.

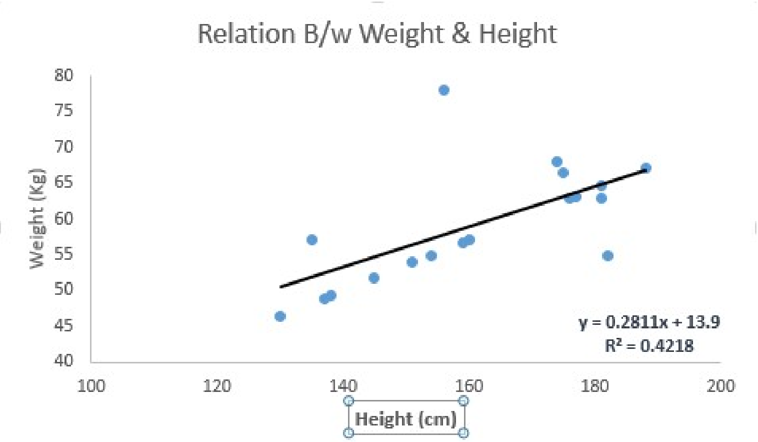

Regression
No of independent variables
Shape of the Regression line
Type of dependent variable

1. Linear Regression

It is one of the most widely known modeling technique. Linear regression is usually among the first few topics which people pick while learning predictive modeling. In this technique, the dependent variable is continuous, independent variable(s) can be continuous or discrete, and nature of regression line is linear.

Linear Regression establishes a relationship between **dependent variable (Y)** and one or more **independent variables (X)** using a **best fit straight line** (also known as regression line).

It is represented by an equation **Y=a+b*X + e**, where a is intercept, b is slope of the line and e is error term. This equation can be used to predict the value of target variable based on given predictor variable(s).



The difference between simple linear regression and multiple linear regression is that, multiple linear regression has (>1) independent variables, whereas simple linear regression has only 1 independent variable.

2. Logistic Regression

Logistic regression is used to find the probability of event=Success and event=Failure. We should use logistic regression when the dependent variable is binary (0/ 1, True/ False, Yes/ No) in nature. Here the value of Y ranges from 0 to 1 and it can represent by following equation.

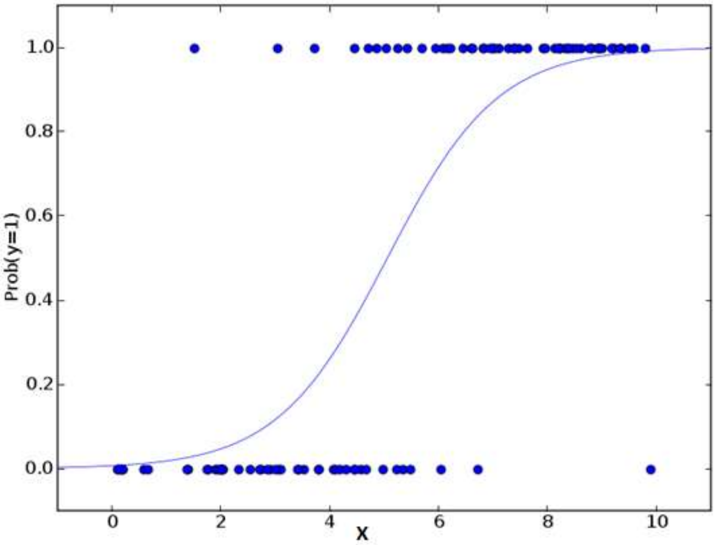
odds= p/ (1-p) = probability of event occurrence / probability of not event occurrence

ln(odds) = ln(p/(1-p))

logit(p) = ln(p/(1-p)) = b0+b1X1+b2X2+b3X3....+bkXk

Above, p is the probability of presence of the characteristic of interest. A question that you should ask here is “why have we used log in the equation?”.

Since we are working here with a binomial distribution (dependent variable), we need to choose a link function which is best suited for this distribution. And, it is **logit** function. In the equation above, the parameters are chosen to maximize the likelihood of observing the sample values rather than minimizing the sum of squared errors (like in ordinary regression).



3. Ridge Regression

Ridge Regression is a technique used when the data suffers from multicollinearity (independent variables are highly correlated).
In multicollinearity, even though the least squares estimates (OLS) are unbiased, their variances are large which deviates the observed value far from the true value. By adding a degree of bias to the regression estimates, ridge regression reduces the standard errors.

Above, we saw the equation for linear regression. Remember? It can be represented as:

$y=a+ b*x$

This equation also has an error term. The complete equation becomes:

y=a+b*x+e (error term), [error term is the value needed to correct for a prediction error between the observed and predicted value]

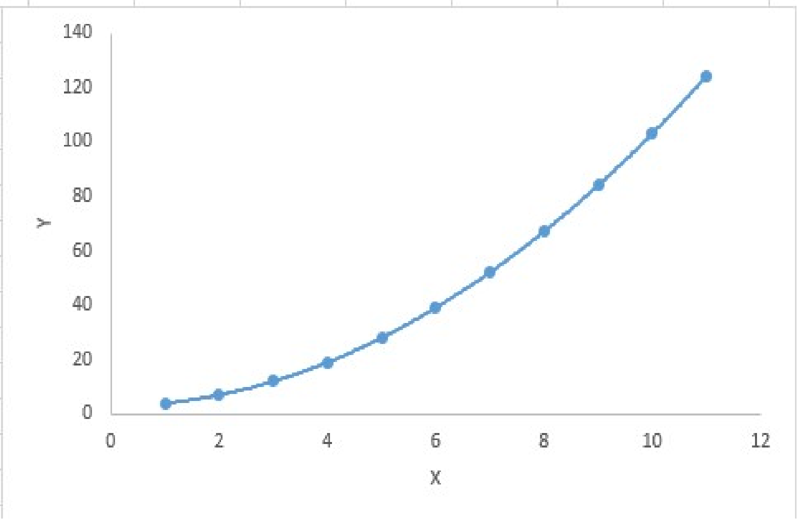
=> y=a+y= a+ b1x1+ b2x2+....+e, for multiple independent variables.

4. Polynomial Regression

A regression equation is a polynomial regression equation if the power of independent variable is more than 1. The equation below represents a polynomial equation:

y=a+b*x^2

In this regression technique, the best fit line is not a straight line. It is rather a curve that fits into the data points.



5. k-nearest neighbors algorithm

In pattern recognition, the ***k*-nearest neighbors algorithm** (***k*-NN**) is a non-parametric method used for classification and regression. In both cases, the input consists of the *k* closest training examples in the feature space. The output depends on whether *k*-NN is used for classification or regression:

- In *k-NN classification*, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors (*k* is a positive integer, typically small). If $k = 1$, then the object is simply assigned to the class of that single nearest neighbor.
- In *k-NN regression*, the output is the property value for the object. This value is the average of the values of *k* nearest neighbors.

k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. The *k*-NN algorithm is among the simplest of all machine learning algorithms.

Both for classification and regression, a useful technique can be used to assign weight to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of $1/d$, where *d* is the distance to the neighbor.

The neighbors are taken from a set of objects for which the class (for *k*-NN classification) or the object property value (for *k*-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required.

A peculiarity of the *k*-NN algorithm is that it is sensitive to the local structure of the data.

Pedestrian detection

is an essential and significant task in any intelligent video surveillance system, as it provides the fundamental information for semantic understanding of the video footages. It has an obvious extension to automotive applications due to the potential for improving safety systems. Many car manufacturers (e.g. Volvo, Ford, GM, Nissan) offer this as an ADAS option in 2017.

Challenges

- Various style of clothing in appearance
- Different possible articulations
- The presence of occluding accessories
- Frequent occlusion between pedestrians

Haar-like features are digital image features used in object recognition. They owe their name to their intuitive similarity with Haar wavelets and were used in the first real-time face detector.[1]

Historically, working with only image intensities (i.e., the RGB pixel values at each and every pixel of image) made the task of feature calculation computationally expensive. A publication by Papageorgiou et al.[2] discussed working with an alternate feature set based on Haar wavelets instead of the usual image intensities. Viola and Jones[1] adapted the idea of using Haar wavelets and developed the so-called Haar-like features. A Haar-like feature considers adjacent rectangular regions at a specific location in a detection window, sums up the pixel intensities in each region and calculates the difference between these sums.

For example, let us say we have an image database with human faces. It is a common observation that among all faces the region of the eyes is darker than the region of the cheeks. Therefore, a common Haar feature for face detection is a set of two adjacent rectangles that lie above the eye and the cheek region. The position of these rectangles is defined relative to a detection window that acts like a bounding box to the target object (the face in this case).

The key advantage of a Haar-like feature over most other features is its calculation speed. Due to the use of *integral images*, a Haar-like feature of any size can be calculated in constant time (approximately 60 microprocessor instructions for a 2-rectangle feature).

Rectangular Haar-like features

A simple rectangular Haar-like feature can be defined as the difference of the sum of pixels of areas inside the rectangle, which can be at any position and scale within the original image. This modified feature set is called *2-rectangle feature*. Viola and Jones also defined 3-rectangle features and 4-rectangle features. The values indicate certain characteristics of a particular area of the image. Each feature type can indicate the existence (or absence) of certain characteristics in the image, such as edges or changes in texture. For example, a 2-rectangle feature can indicate where the border lies between a dark region and a light region.

Abstract – Robust and efficient vehicle detection from images is an important task in Intelligent Transportation Systems. With the development of computer vision techniques and consequent accessibility of video image data, new applications have been enabled to

on-road vehicle detection algorithms

Vehicle Detection Approaches

To be useful, vehicle detection methods need to be fast enough to operate in real-time, be insensitive to illumination change and different weather conditions, and be able to separate vehicles from images sequences in an accurate and

efficient manner. With the deployment of video cameras, vehicle detection can be categorized as feature-based methods and model-based methods.

Vehicle Detection Approaches

To be useful, vehicle detection methods need to be fast enough to operate in real-time, be insensitive to illumination change and different weather conditions, and be able to separate vehicles from images sequences in an accurate and efficient manner. With the deployment of video cameras, vehicle detection can be categorized as feature-based methods and model-based methods.

Pedestrian and Car detection

The aim of this project is to develop an application which can detect pedestrians effectively. The problem of motion-based object detection can be divided into two parts:
a) Classifying pedestrians and non pedestrians features
a) Detecting pedestrians in each frame
b) Associating the detections corresponding to the same object over time

Tool: This project is based on Machine learning, We can provide image data set of pedestrians and non-pedestrians as an training data to the software tool which will extract important features using An object detection method that inputs Haar features into a series of classifiers (cascade) to identify objects in an image. We can use Python or Machine Learning as a building tool for this system.

Implementation : The Implementation of such a tool depends on two factors – Feature extraction and object detection methods. So we use HAAR classifiers and also read about basic feature extraction algorithm.

Research: Detecting humans in images is a challenging task owing to their variable appearance. This is a booming research topic which is still going on for surveillance of large crowds in real time applications. Research areas include image processing, artificial Intelligence and machine learning.

The objective of the program given is to detect object of interest (Car) in video frames and to keep tracking the same object. This is an example of how to detect vehicles in Python.

Why Vehicle Detection?

- The startling losses both in human lives and finance caused by vehicle accidents.
- Detecting vehicles in images acquired from a moving platform is a challenging problem.

Haar Cascades for Human Detection

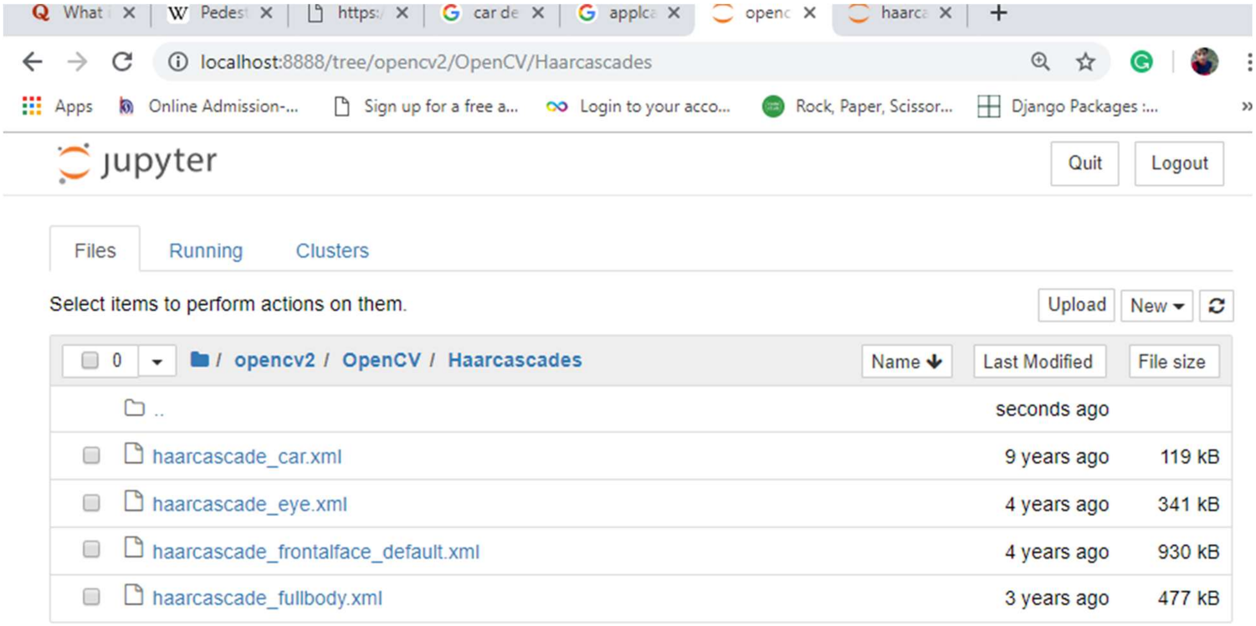
Haar feature based approach for object detection is proposed by Paul Viola and Michael Jones in their paper "Rapid Object Detection using a Boosted Cascade of Simple Features" published in 2001. This approach is widely used for Face Detection.

OpenCV includes inbuilt functionality to provide Haar cascade-based object detection. Pre-trained models provided by OpenCV for "Full Body Detection", "Upper Body Detection" and "Lower Body Detection" are available here.

This Python code snippet shows application of Haar cascade for Human Detection using Open CV 3.4. It shows a frame time of approximately 90 — 100 milliseconds per frame (equivalent to 11 frames-per-second) in my test bench.

<u>Screenshot of project</u>

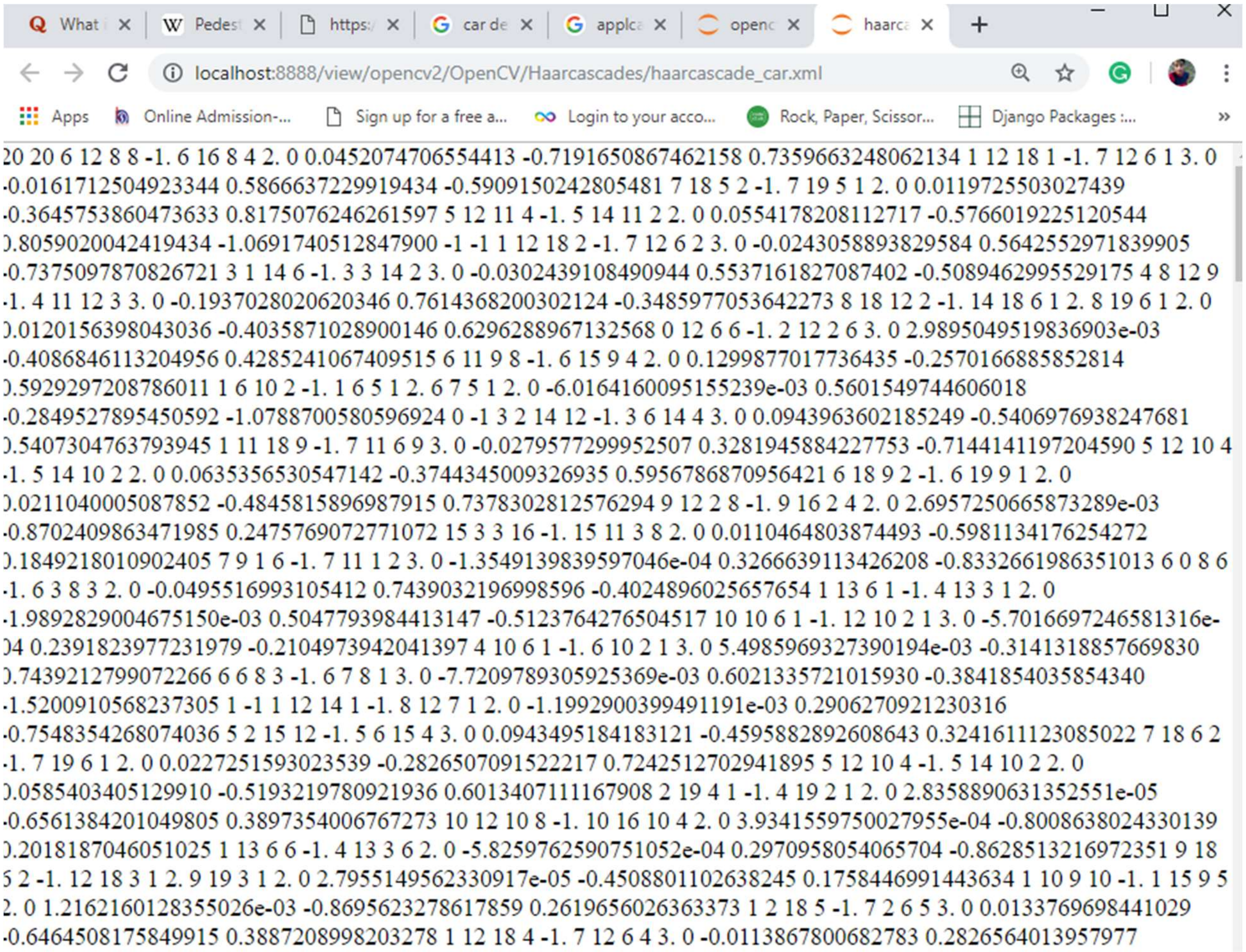
Haar cascade: DIR



Select items to perform actions on them.

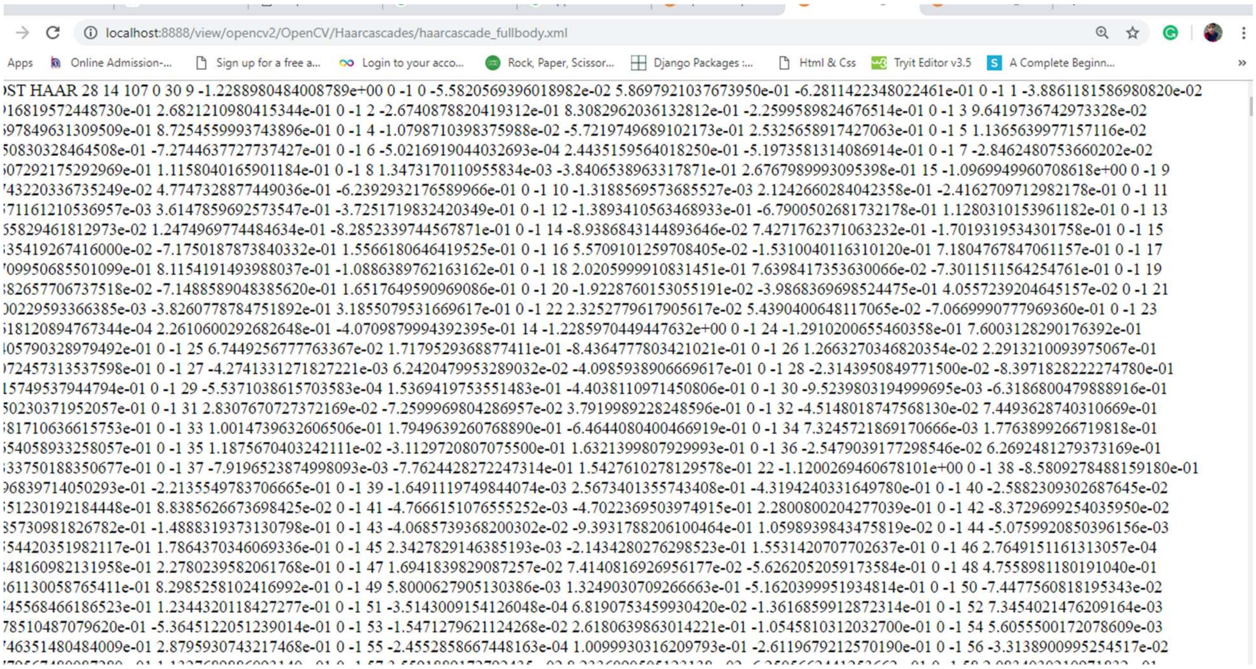
/ opencv2 / OpenCV / Haarcascades

Name	Last Modified	File size
..	seconds ago	
haarcascade_car.xml	9 years ago	119 kB
haarcascade_eye.xml	4 years ago	341 kB
haarcascade_frontalface_default.xml	4 years ago	930 kB
haarcascade_fullbody.xml	3 years ago	477 kB



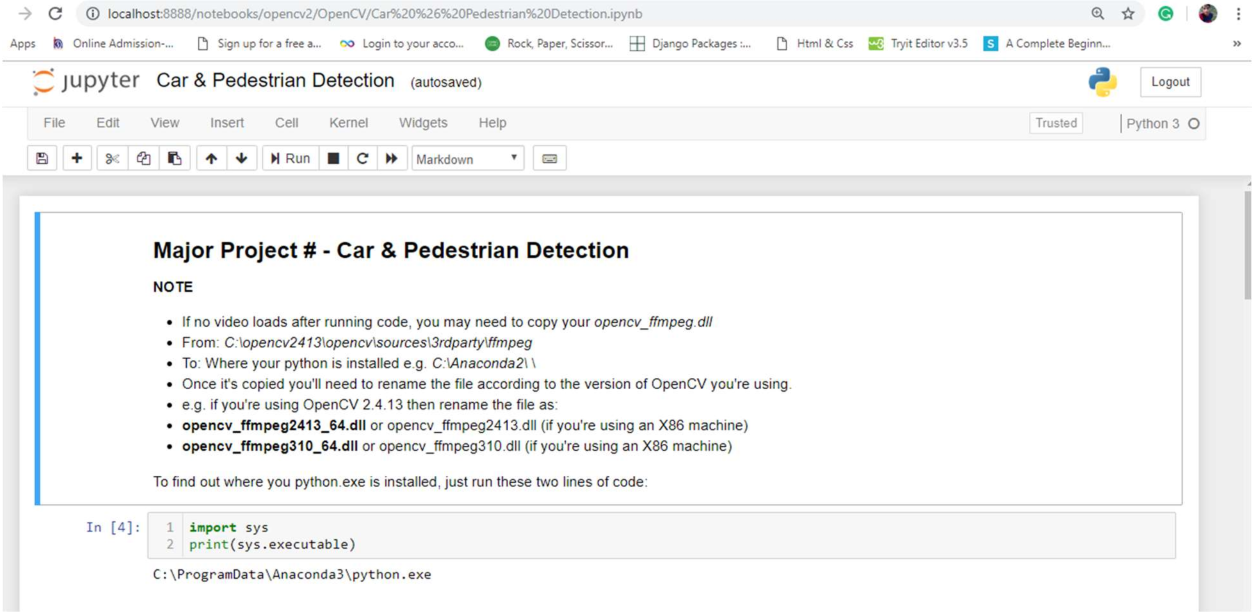
```
20 20 6 12 8 8 -1. 6 16 8 4 2. 0 0.0452074706554413 -0.7191650867462158 0.7359663248062134 1 12 18 1 -1. 7 12 6 1 3. 0
-0.0161712504923344 0.5866637229919434 -0.5909150242805481 7 18 5 2 -1. 7 19 5 1 2. 0 0.0119725503027439
-0.3645753860473633 0.8175076246261597 5 12 11 4 -1. 5 14 11 2 2. 0 0.0554178208112717 -0.5766019225120544
0.8059020042419434 -1.0691740512847900 -1 -1 1 12 18 2 -1. 7 12 6 2 3. 0 -0.0243058893829584 0.5642552971839905
-0.7375097870826721 3 1 14 6 -1. 3 3 14 2 3. 0 -0.0302439108490944 0.5537161827087402 -0.5089462995529175 4 8 12 9
-1. 4 11 12 3 3. 0 -0.1937028020620346 0.7614368200302124 -0.3485977053642273 8 18 12 2 -1. 14 18 6 1 2. 8 19 6 1 2. 0
0.0120156398043036 -0.4035871028900146 0.6296288967132568 0 12 6 6 -1. 2 12 2 6 3. 0 2.9895049519836903e-03
-0.4086846113204956 0.4285241067409515 6 11 9 8 -1. 6 15 9 4 2. 0 0.1299877017736435 -0.2570166885852814
0.5929297208786011 1 6 10 2 -1. 1 6 5 1 2. 6 7 5 1 2. 0 -6.0164160095155239e-03 0.5601549744606018
-0.2849527895450592 -1.0788700580596924 0 -1 3 2 14 12 -1. 3 6 14 4 3. 0 0.0943963602185249 -0.5406976938247681
0.5407304763793945 1 11 18 9 -1. 7 11 6 9 3. 0 -0.0279577299952507 0.3281945884227753 -0.7144141197204590 5 12 10 4
-1. 5 14 10 2 2. 0 0.0635356530547142 -0.3744345009326935 0.5956786870956421 6 18 9 2 -1. 6 19 9 1 2. 0
0.0211040005087852 -0.4845815896987915 0.7378302812576294 9 12 2 8 -1. 9 16 2 4 2. 0 2.6957250665873289e-03
-0.8702409863471985 0.2475769072771072 15 3 3 16 -1. 15 11 3 8 2. 0 0.0110464803874493 -0.5981134176254272
0.1849218010902405 7 9 1 6 -1. 7 11 1 2 3. 0 -1.3549139839597046e-04 0.3266639113426208 -0.8332661986351013 6 0 8 6
-1. 6 3 8 3 2. 0 -0.0495516993105412 0.7439032196998596 -0.4024896025657654 1 13 6 1 -1. 4 13 3 1 2. 0
-1.9892829004675150e-03 0.5047793984413147 -0.5123764276504517 10 10 6 1 -1. 12 10 2 1 3. 0 -5.7016697246581316e-
04 0.2391823977231979 -0.2104973942041397 4 10 6 1 -1. 6 10 2 1 3. 0 5.4985969327390194e-03 -0.3141318857669830
0.7439212799072266 6 6 8 3 -1. 6 7 8 1 3. 0 -7.7209789305925369e-03 0.6021335721015930 -0.3841854035854340
-1.5200910568237305 1 -1 1 12 14 1 -1. 8 12 7 1 2. 0 -1.1992900399491191e-03 0.2906270921230316
-0.7548354268074036 5 2 15 12 -1. 5 6 15 4 3. 0 0.0943495184183121 -0.4595882892608643 0.3241611123085022 7 18 6 2
-1. 7 19 6 1 2. 0 0.0227251593023539 -0.2826507091522217 0.7242512702941895 5 12 10 4 -1. 5 14 10 2 2. 0
0.0585403405129910 -0.5193219780921936 0.6013407111167908 2 19 4 1 -1. 4 19 2 1 2. 0 2.8358890631352551e-05
-0.6561384201049805 0.3897354006767273 10 12 10 8 -1. 10 16 10 4 2. 0 3.9341559750027955e-04 -0.8008638024330139
0.2018187046051025 1 13 6 6 -1. 4 13 3 6 2. 0 -5.8259762590751052e-04 0.2970958054065704 -0.8628513216972351 9 18
5 2 -1. 12 18 3 1 2. 9 19 3 1 2. 0 2.7955149562330917e-05 -0.4508801102638245 0.1758446991443634 1 10 9 10 -1. 1 15 9 5
2. 0 1.2162160128355026e-03 -0.8695623278617859 0.2619656026363373 1 2 18 5 -1. 7 2 6 5 3. 0 0.0133769698441029
-0.6464508175849915 0.3887208998203278 1 12 18 4 -1. 7 12 6 4 3. 0 -0.0113867800682783 0.2826564013957977
```

XML for car detection



```
ST HAAR 28 14 107 0 30 9 -1.2288980484008789e+00 0 -1 0 -5.5820569396018982e-02 5.8697921037673950e-01 -6.2811422348022461e-01 0 -1 1 -3.8861181586980820e-02
16819572448730e-01 2.6821210980415344e-01 0 -1 2 -2.6740878820419312e-01 8.3082962036132812e-01 -2.2599589824676514e-01 0 -1 3 9.6419736742973328e-02
97849631309509e-01 8.7254559993743896e-01 0 -1 4 -1.0798710398375988e-02 -5.7219749689102173e-01 2.5325658917427063e-01 0 -1 5 1.1365639977157116e-02
50830328464508e-01 -7.2744637727737427e-01 0 -1 6 -5.0216919044032693e-04 2.4435159564018250e-01 -5.1973581314086914e-01 0 -1 7 -2.8462480753660202e-02
07292175292969e-01 1.1158040165901184e-01 0 -1 8 1.3473170110955834e-03 -3.8406538963317871e-01 2.6767989993095398e-01 15 -1.0969949960708618e+00 0 -1 9
43220336735249e-02 4.7747328877449036e-01 -6.2392932176589966e-01 0 -1 10 -1.3188569573685527e-03 2.1242660284042358e-01 -2.4162709712982178e-01 0 -1 11
71161210536957e-03 3.6147859692573547e-01 -3.7251719832420349e-01 0 -1 12 -1.3893410563468933e-01 -6.7900502681732178e-01 1.1280310153961182e-01 0 -1 13
55829461812973e-02 1.2474969774484634e-01 -8.2852339744567871e-01 0 -1 14 -8.9386843144893646e-02 7.4271762371063232e-01 -1.7019319534301758e-01 0 -1 15
35419267416000e-02 -7.1750187873840332e-01 1.5566180646419525e-01 0 -1 16 5.5709101259708405e-02 -1.5310040116310120e-01 7.1804767847061157e-01 0 -1 17
09950685501099e-01 8.1154191493988037e-01 -1.0886389762163162e-01 0 -1 18 2.0205999910831451e-01 7.6398417353630066e-02 -7.3011511564254761e-01 0 -1 19
82657706737518e-02 -7.1488589048385620e-01 1.6517649590969086e-01 0 -1 20 -1.9228760153055191e-02 -3.9868369698524475e-01 4.0557239204645157e-02 0 -1 21
02295933663385e-03 -3.8260778784751892e-01 3.1855079531669617e-01 0 -1 22 2.3252779617905617e-02 5.4390400648117065e-02 -7.0669990777969360e-01 0 -1 23
18120894767344e-04 2.2610600292682648e-01 -4.0709879994392395e-01 14 -1.2285970449447632e+00 0 -1 24 -1.2910200655460358e-01 7.6003128290176392e-01
05790328979492e-01 0 -1 25 6.7449256777763367e-02 1.7179529368877411e-01 -8.4364777803421021e-01 0 -1 26 1.2663270346820354e-02 2.2913210093975067e-01
72457313537598e-01 0 -1 27 -4.2741331271827221e-03 6.2420479953289032e-02 -4.0985938906669617e-01 0 -1 28 -2.3143950849771500e-02 -8.3971828222274780e-01
15749537944794e-01 0 -1 29 -5.5371038615703583e-04 1.5369419753551483e-01 -4.4038110971450806e-01 0 -1 30 -9.5239803194999695e-03 -6.3186800479888916e-01
50230371952057e-01 0 -1 31 2.8307670727372169e-02 -7.2599969804286957e-02 3.7919989228248596e-01 0 -1 32 -4.5148018747568130e-02 7.4493628740310669e-01
81710636615753e-01 0 -1 33 1.0014739632606506e-01 1.7949639260768890e-01 -6.4644080400466919e-01 0 -1 34 7.3245721869170666e-03 1.7763899266719818e-01
54058933258057e-01 0 -1 35 1.1875670403242111e-02 -3.1129720807075500e-01 1.6321399807929993e-01 0 -1 36 -2.5479039177298546e-02 6.2692481279373169e-01
33750188350677e-01 0 -1 37 -7.9196523874998093e-03 -7.7624428272247314e-01 1.5427610278129578e-01 22 -1.1200269460678101e+00 0 -1 38 -8.5809278488159180e-01
68397140502093e-01 -2.2135549783706665e-01 0 -1 39 -1.6491119749844074e-03 2.5673401355743408e-01 -4.3194240331649780e-01 0 -1 40 -2.5882309302687645e-02
51230192184448e-01 8.8385626673698425e-02 0 -1 41 -4.7666151076555252e-03 -4.7022369503974915e-01 2.2800800204277039e-01 0 -1 42 -8.3729699254035950e-02
57309818267820e-01 -1.4888319373130798e-01 0 -1 43 -4.0685739368200302e-02 -9.3931788206100464e-01 1.0598939843475819e-02 0 -1 44 -5.0759920850396156e-03
54420351982117e-01 1.7864370346069336e-01 0 -1 45 2.3427829146385193e-03 -2.1434280276298523e-01 1.5531420707702637e-01 0 -1 46 2.7649151161313057e-04
48160982131958e-01 2.2780239582061768e-01 0 -1 47 1.6941839829087257e-02 7.4140816926956177e-02 -5.6262052059173584e-01 0 -1 48 4.7558981180191040e-01
61130058765411e-01 8.2985258102416992e-01 0 -1 49 5.8000627905130386e-03 1.3249030709266663e-01 -5.1620399951934814e-01 0 -1 50 -7.4477560818195343e-02
45568466186523e-01 1.2344320118427277e-01 0 -1 51 -3.5143009154126048e-04 6.8190753459930420e-02 -1.3616859912872314e-01 0 -1 52 7.3454021476209164e-03
78510487079620e-01 -5.3645122051239014e-01 0 -1 53 -1.5471279621124268e-02 2.6180639863014221e-01 -1.0545810312032700e-01 0 -1 54 5.6055500172078609e-03
46351480484009e-01 2.8795930743217468e-01 0 -1 55 -2.4552858667448163e-04 1.0099930316209793e-01 -2.6119679212570190e-01 0 -1 56 -3.3138900995254517e-02
```

Project source code:



Major Project # - Car & Pedestrian Detection

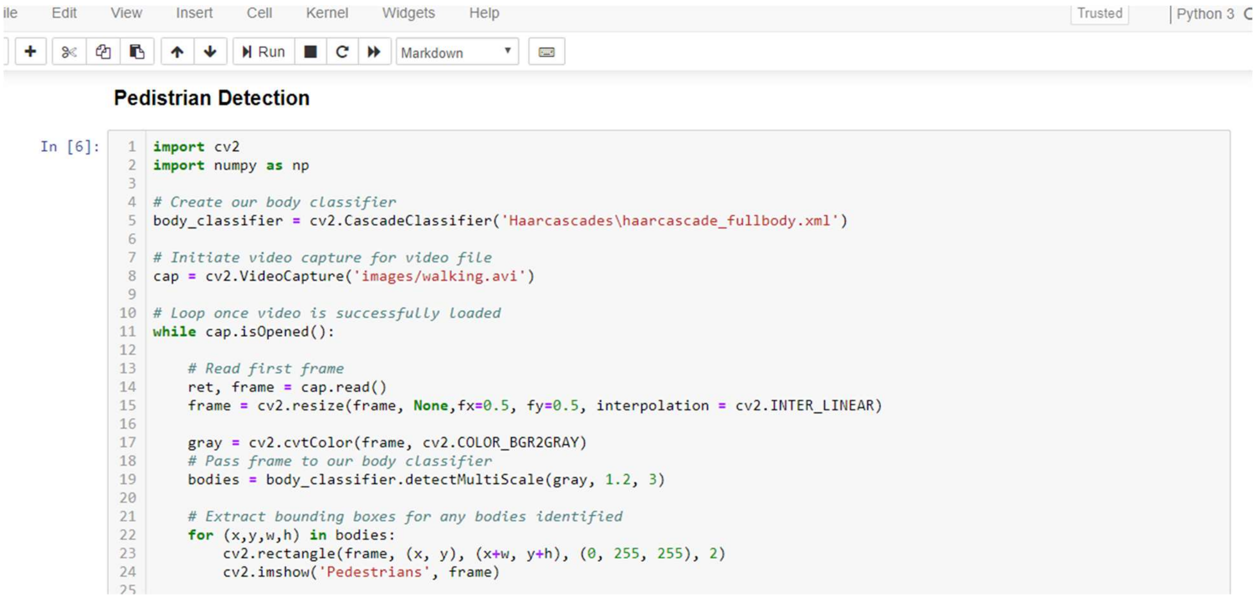
NOTE

- If no video loads after running code, you may need to copy your *opencv_ffmpeg.dll*
- From: *C:\opencv2413\opencv\sources\3rdparty\ffmpeg*
- To: Where your python is installed e.g. *C:\Anaconda2\* \
- Once it's copied you'll need to rename the file according to the version of OpenCV you're using.
- e.g. if you're using OpenCV 2.4.13 then rename the file as:
- **opencv_ffmpeg2413_64.dll** or opencv_ffmpeg2413.dll (if you're using an X86 machine)
- **opencv_ffmpeg310_64.dll** or opencv_ffmpeg310.dll (if you're using an X86 machine)

To find out where you python.exe is installed, just run these two lines of code:

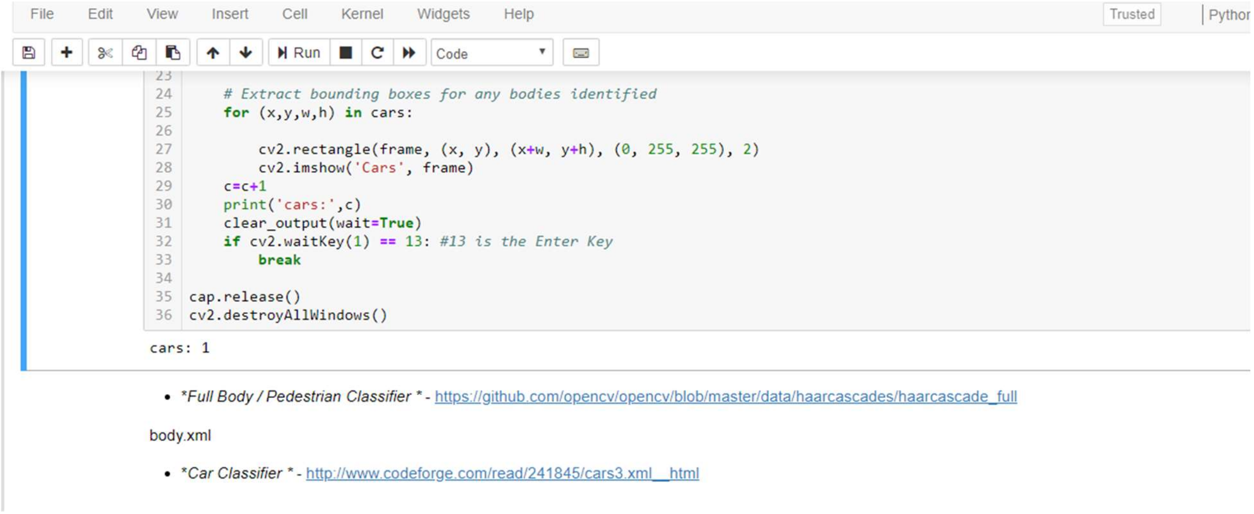
```
In [4]:
import sys
print(sys.executable)
```

```
C:\ProgramData\Anaconda3\python.exe
```



Pedestrian Detection

```python
import cv2
import numpy as np

# Create our body classifier
body_classifier = cv2.CascadeClassifier('Haarcascades\haarcascade_fullbody.xml')

# Initiate video capture for video file
cap = cv2.VideoCapture('images/walking.avi')

# Loop once video is successfully loaded
while cap.isOpened():

    # Read first frame
    ret, frame = cap.read()
    frame = cv2.resize(frame, None,fx=0.5, fy=0.5, interpolation = cv2.INTER_LINEAR)

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    # Pass frame to our body classifier
    bodies = body_classifier.detectMultiScale(gray, 1.2, 3)

    # Extract bounding boxes for any bodies identified
    for (x,y,w,h) in bodies:
        cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 255), 2)
        cv2.imshow('Pedestrians', frame)
```



```python

    # Extract bounding boxes for any bodies identified
    for (x,y,w,h) in cars:

        cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 255), 2)
        cv2.imshow('Cars', frame)
    c=c+1
    print('cars:',c)
    clear_output(wait=True)
    if cv2.waitKey(1) == 13: #13 is the Enter Key
        break

cap.release()
cv2.destroyAllWindows()
```

```
cars: 1
```

- *Full Body / Pedestrian Classifier * - https://github.com/opencv/opencv/blob/master/data/haarcascades/haarcascade_full

body.xml

- *Car Classifier * - http://www.codeforge.com/read/241845/cars3.xml__html