

**A**  
**PROJECT REPORT**  
**on**  
**YOUTUBE SPAM COMMENTS DETECTION**

**Submitted in partial fulfilment for the Award of Degree of**  
**BACHELOR OF ENGINEERING**  
**IN**  
**INFORMATION TECHNOLOGY**

**BY**  
**BODIGE ARAVIND KUMAR (160117737033)**  
**&**  
**NALLA ARUN REDDY (160117737035)**

Under the guidance of

**Ms. A.Sirisha**  
Assistant Professor,  
Dept. of IT, CBIT



**DEPARTMENT OF INFORMATION TECHNOLOGY**  
**CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY (A)**

(Affiliated to Osmania University; Accredited by NBA (AICTE) and NAAC (UGC), ISO  
Certified 9001:2015), Kokapet (V), GANDIPET(M), HYDERABAD – 500 075

Website: [www.cbit.ac.in](http://www.cbit.ac.in)

**2020-2021**



---

## **CERTIFICATE**

This is to certify that the project work entitled **“YOUTUBE SPAM COMMENTS DETECTION”** submitted by **BODIGE ARAVIND KUMAR (160117737033)** and **NALLA ARUN REDDY(160117737035)** in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF ENGINEERING in INFORMATION TECHNOLOGY** to **CHAITANYA BHARATHI INSTITUTE OF TECHNOLOGY(A)**, affiliated to **OSMANIA UNIVERSITY**, Hyderabad, is a record of bonafide work carried out by them under my supervision and guidance. The results embodied in this report have not been submitted to any other University or Institution for the award of any other Degree or Diploma.

### **Project Guide**

**Ms. A.Sirisha**

Asst. Professor, Dept. Of IT,  
CBIT, Hyderabad.

### **Head of the Department**

**Dr. K. Radhika**

Professor & Head, Dept. Of IT,  
CBIT, Hyderabad.

## **DECLARATION**

We declare that the project report entitled “**YOUTUBE SPAM COMMENTS DETECTION**” is being submitted by us in the Department of Information Technology, Chaitanya Bharathi Institute of Technology (A), Osmania University.

This is record of bonafide work carried out by us under the guidance and supervision of **Ms. A. Sirisha**, Assistant Professor, Dept. of IT, C.B.I.T.

No part of the work is copied from books/journals/internet and wherever the portion is taken, the same has been duly referred in the text. The report is based on the project work done entirely by us and not copied from any other source.

**Bodige Aravind Kumar (160117737033)**

**Nalla Arun Reddy (160117737035)**

## ACKNOWLEDGEMENTS

It is our privilege to acknowledge with deep sense of gratitude and devotion for keen personal interest and invaluable guidance rendered by our Project Guide **Ms. A.Sirisha**, Assistant Professor, Department of Information, Chaitanya Bharathi Institute of Technology.

We take the opportunity to express our thanks to **Dr K. Radhika**, Professor & Head of IT Department, CBIT for her valuable suggestions and moral support.

We are grateful to our Principal **Dr. P. Ravinder Reddy**, Chaitanya Bharathi Institute of Technology, for his cooperation and encouragement.

Finally, we also thank all the staff members, faculty of Dept. of IT, CBIT, and our friends, who with their valuable suggestions and support, directly or indirectly helped us in completing this project work.

## **ABSTRACT**

People now feel more comfortable socializing over the internet through popular social networking and media websites than face to face. Thus, the social media websites are thriving more and more nowadays. Like others YouTube is a vastly popular social media site which is expanding at very fast pace. YouTube depends mostly on user created contents sharing and spreading.

However, due to this popularity, YouTube has become more susceptible to different types of unwanted and malicious spammers. With the raised quality of online social networks, spammers realize these platforms are simple to lure users into malicious activities by posting spam messages in the comments section of the videos.

This project classifies whether the youtube comments are legitimate or spam comments. In this project first some sample datasets are taken and trained on different models. Each model classifies spam and non spam comments. An optimal model is being chosen (high accuracy model) and is deployed into web application. This web application takes a YouTube video link as input and detects spam comments in it. The accuracy of model is been observed as 95%.

## **TABLE OF CONTENTS**

<b>TITLE</b>	<b>Page No</b>
<b>CERTIFICATE</b>	<b>ii</b>
<b>DECLARATION</b>	<b>iii</b>
<b>ACKNOWLEDGEMENTS</b>	<b>iv</b>
<b>ABSTRACT</b>	<b>v</b>
<b>LIST OF FIGURES</b>	<b>viii</b>
<b>LIST OF TABLES</b>	<b>ix</b>
<b>LIST OF ABBREVIATIONS</b>	<b>x</b>
<b>1. INTRODUCTION</b>	<b>1</b>
1.1 PROBLEM STATEMENT	1
1.2 APPLICATIONS	1
1.3 ORGANIZATION OF REPORT	2
<b>2. LITERATURE SURVEY</b>	<b>3</b>
2.1 PAPER 1	3
2.2 PAPER 2	5
2.3 PAPER 3	7
<b>3. SYSTEM REQUIREMENT SPECIFICATIONS</b>	<b>9</b>
3.1 FUNCTIONAL REQUIREMENTS	9
3.2 NON-FUNCTIONAL REQUIREMENTS	9
3.3 SOFTWARE REQUIREMENTS	10
3.3.1 GOOGLE COLAB	10
3.3.2 PYCHARM	10
3.3.3 FLASK	11
3.4 HARDWARE REQUIREMENTS	12
<b>4. METHODOLOGY</b>	<b>13</b>

4.1 ARCHITECTURE	14
<b>5. IMPLEMENTATION</b>	<b>20</b>
<b>6. TESTING AND RESULTS</b>	<b>28</b>
<b>7. CONCLUSION AND FUTURE SCOPE</b>	<b>33</b>
<b>BIBLIOGRAPHY</b>	<b>34</b>

## LIST OF FIGURES

<b>No</b>	<b>NAME OF FIGURE</b>	<b>Page No</b>
Figure 2.1.	Diagram of Markov decision process	4
Figure 2.2	Graph of SVM model	6
Figure 2.3	N-Gram Analysis graph	8
Figure 4.1	Sequential Flow of operations	14
Figure 4.2	A typical example for spam comment on youtube	15
Figure 5.1	Python installation page	20
Figure 5.2	Pycharm setup page	21
Figure 5.3	Pycharm welcome page	22
Figure 5.4	Pycharm project create page	22
Figure 5.5	FLASK application running on local host	23
Figure 5.6	Essential packages for project	23
Figure 5.7	Web scrapping of youtube comments	24
Figure 5.8	Prediction and classification of comments	25
Figure 5.9	Logistic regression for comparative analysis	26
Figure 5.10	Naive for comparative analysis	26
Figure 5.11	SVM for comparative analysis	26
Figure 5.12	Random forest classifier for comparative analysis	27
Figure 6.1	Splitting the whole data into train and test sets	28
Figure 6.2	Classification report for Logistic regression	28
Figure 6.3	Classification report for Naïve Bayes	29
Figure 6.4	Classification report for Random forest classifier	29
Figure 6.5	Classification report for SVM	30
Figure 6.6	First page on local host	31
Figure 6.7	Comments scraping on other chrome browser	31
Figure 6.8	Spam comments	32
Figure 6.9	Ham comments	32



## LIST OF TABLES

<b>No</b>	<b>Name of Table</b>	<b>Page No.</b>
Table 3.1	Software requirements	10
Table 3.2	Laptop/PC Hardware Specifications	12
Table 4.1	Datasets Used	15
Table 6.1	Accuracies for different test sizes and various models	30

## **LIST OF ABBREVIATIONS**

NLP-Natural Language Processing

IDE- Integrated Development Environment

WSGI-Web Server Gateway Interface

CART-Classification and Regression Trees

SVM-Super Vector Machine

# **1. INTRODUCTION**

Social networking websites have become an integral part of the day to day lives of people. People turn to social media for interacting with other people, sharing ideas, gaining knowledge, for entertainment and staying informed about the events happening in the rest of the world. Among these sites, YouTube has emerged as the most popular website for sharing and viewing video content. This popularity of YouTube has also attracted spammers, who upload videos with the sole purpose of polluting the system content and causing dissatisfaction among other viewers. These spam videos may be unrelated to their title or may contain pornographic content. Therefore, it is very important to find a way to detect these videos and report them before they are viewed by innocent users.

## **1.1 PROBLEM STATEMENT**

Youtube themselves have been blocking the comments like urls in the comment section. Such methods have proven to be extremely ineffective as spammers have found ways to by pass such heuristics. Standard machine learning classification algorithms have proven to be somewhat effective but there is still room for better accuracy with new approaches. To design a model which detects youtube spam comments in real time using machine learning classification algorithms by taking youtube video URL as input.

## **1.2 APPLICATIONS**

This technology can be leveraged for various purposes and in fields. As every process these days is being automated here are a few applications of this project idea.

Among different kind of undesired content, YouTube is facing problems to manage the huge volume of undesired text comments posted by users that aim to self-promote their videos, or to disseminate malicious links to steal private data. To stop such kind of activities this spam detection can be supportive.

### **1.3 ORGANIZATION OF THE REPORT**

**Chapter 1** deals with the introduction of the project and explains the purpose of the project.

**Chapter 2** deals with the literature survey

**Chapter 3** deals with the requirements that are needed in order to execute the project

**Chapter 4** deals with the methodology, system design and features of the project

**Chapter 5** deals with the implementation of the project

**Chapter 6** discuss the results of the project

**Chapter 7** involves conclusion and future scope of the project.

## **2. LITERATURE SURVEY**

Based on the problem statement and the project requirements the following research papers have been referred.

### **2.1. PAPER 1**

**Title** – A Novel Approach for Youtube Video Spam Detection using Markov Decision Process

**Authors:** Simran Kanodia, Rachna Sasheendran, Vinod Pathari

**Year of publication:** 2018

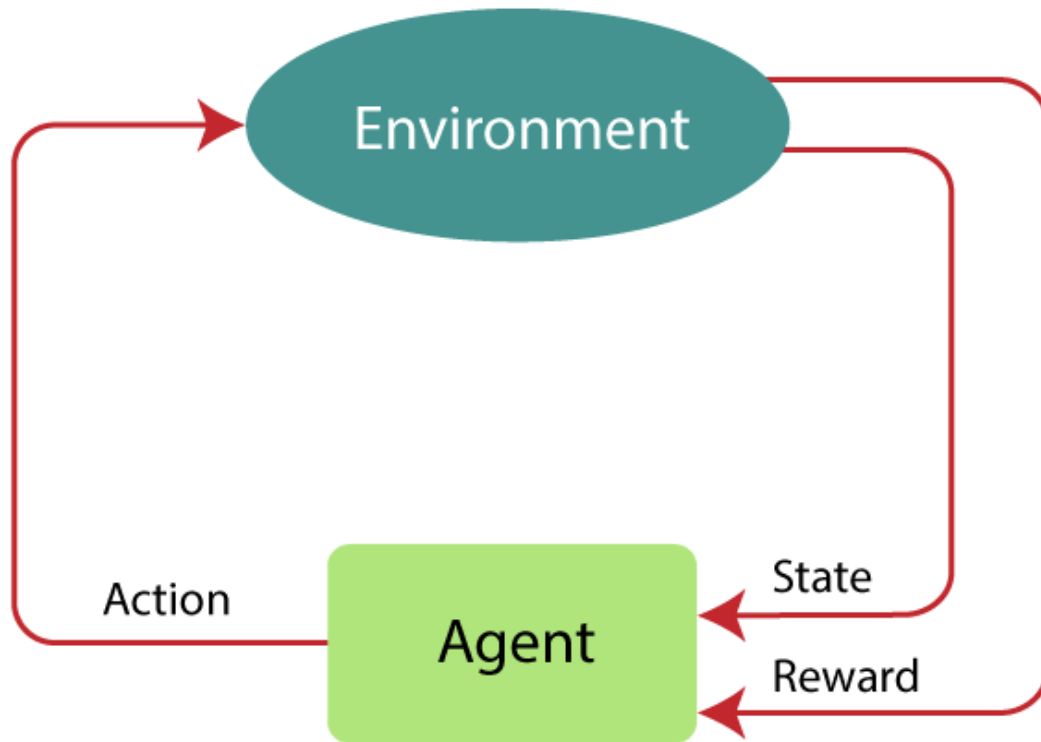
**Publisher name:** IEEE

#### **2.1.1 METHODOLOGY**

This paper used markov decision process to detect the YouTube spam comments. This model gave approximately 78.8 % accuracy

A Markov Decision Process (MDP) is a mathematical model that is used to make decisions in a stochastic environment. A stochastic environment is one where outcomes are partially random and partially under the decision maker's control.

In an MDP, the environment is modeled as a discrete-time state-transition system with a set of states and actions. The states represent the outcomes of the decision-making process, and an action represents a decision which can be taken from a particular state. The agent receives a reward as a consequence of the decision it chooses from a particular state . The agent aims to maximize the total reward of the model calculated over all the states.



**Fig 2.1. Diagram of Markov decision process**

#### **A. Datasets**

For data generation, this paper used publicly available APIs to crawl YouTube and extract the required attributes of a given video.

#### **B. Advantages**

- It takes real time youtube video link and evaluate results

#### **C. Drawback**

- Low accuracy 78.8 %

## **2.2 PAPER 2**

**Title -** A Comparative Analysis of Common YouTube Comment Spam Filtering Techniques

**Authors:** Abdullah O. Abdullah, Mashhood A. Ali, Murat Karabatak, Abdulkadir Sengur

**Year of publication:** 2018

**Publisher name:** IEEE

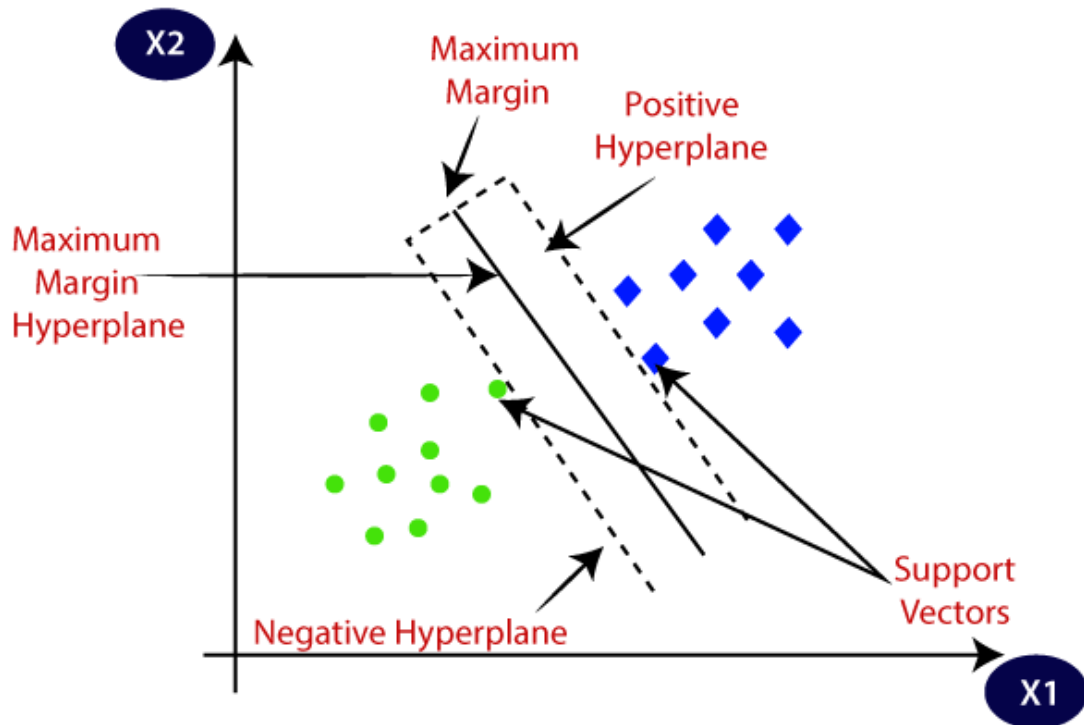
### **2.2.1 METHODOLOGY**

In this paper the weka tool is used, the author used Super Vector Machine model for testing and training the datasets.

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.



**Fig 2.2. Graph of SVM model**

#### **A. Datasets**

The datasets used in this project from YouTube using YouTube Data API .

#### **B. Advantages**

- Easy implementation
- Better accuracy

#### **C. Drawback**

- It can not be used in real time analysis by taking youtube links



## **2.3 PAPER 3**

**Title** – N-Gram Assisted Youtube Spam Comment Detection

**Authors:** Shreyas Aiyar ,Nisha P Shetty

**Year of publication:** 2018

**Publisher name:** Scopus

### **2.3.1 METHODOLOGY**

An N-gram language model predicts the probability of a given N-gram within any sequence of words in the language. If we have a good N-gram model, we can predict  $p(w | h)$  – what is the probability of seeing the word  $w$  given a history of previous words  $h$  – where the history contains  $n-1$  words

#### **A. Naive Bayes Classifier**

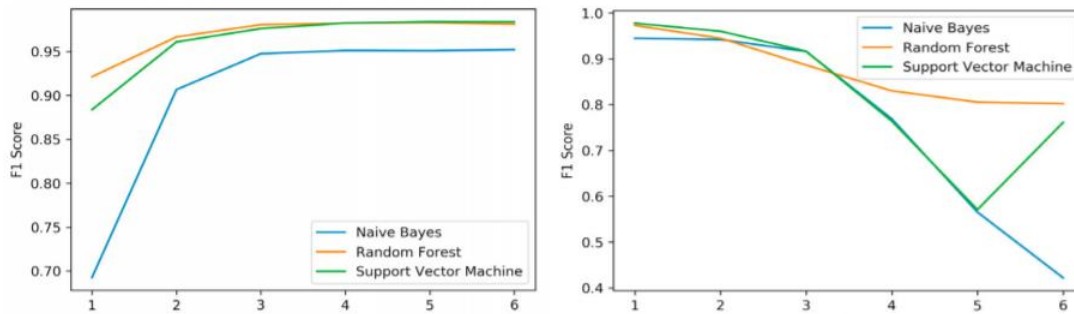
- Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
- It is mainly used in text classification that includes a high-dimensional training dataset..Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.

#### **B. Random forest classifier**

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

### C. SVM model

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.



**Fig 2.3 N-Gram Analysis graph**

### D. Datasets

This project manually extracted around 13000 comments from various channels across Youtube using the public Youtube API

### E. Advantages

- Usage of Hybrid model
- This project works on real time analysis

### F. Drawbacks

- No web application is implemented

### **3. SYSTEM REQUIREMENTS SPECIFICATION**

#### **3.1 FUNCTIONAL REQUIREMENTS**

For a successful project, the requirements must be clear. Especially the evaluation metric must be specified. In ML systems, the quality of the resulting predictions can be considered a functional requirement. We consider predictive power as functional requirement. Our literature survey mentioned measures to quantify the predictive power including accuracy, precision, and recall. The customers don't understand the performance measures. Selecting and interpreting performance measures appropriately is crucial for the acceptance of ML systems. Quantification of quality targets is certainly also a challenge for conventional software, but training an ML model to go beyond a certain utility breakpoint turns into a functional requirement in practice. Predictive power of our model will be greater than 90%. Our model will be deployed as web application, whenever the user gives the youtube video link the model will detect spam comments.

#### **3.2 NON-FUNCTIONAL REQUIREMENTS**

Performance and response time must be good for a good ML model. Along with the performance accuracy also need to be good. Calculation time and response time should be as little as possible, because one of the software's features is timesaving. Whole cycle should not be more than 30 seconds. The system should be easy to use. As we deploy this model as extension it will be easy for user to use it.

### 3.3 SOFTWARE REQUIREMENTS

**Table 3.1. Software Requirements**

S no.	Software	Description
1.	Operating System	Microsoft Windows 10
2.	Integrated Development Environment	Pycharm
3.	Programming language	Python
4.	Framework	Flask
5.	Server	localhost
6.	Tools	HTML, CSS

#### 3.3.1 Google collab

Collaboratory, or “Collab” for short, is a product from Google Research. Collab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Collab is a hosted Jupyter notebook service that requires no setup to use, while providing free access to computing resources including GPUs

If user want to create a machine learning model but say user don't have a computer that can take the workload, Google Colab is the platform for user. Even if user have a GPU or a good computer creating a local environment with anaconda and installing packages and resolving installation issues are a hassle.

Colaboratory is a free Jupyter notebook environment provided by Google where user can use free GPUs and TPUs which can solve all these issues.

#### 3.3.2 Pycharm

PyCharm is a hybrid-platform developed by JetBrains as an IDE for Python. It is commonly used for Python application development. Some of the unicorn organizations such as Twitter, Facebook, Amazon, and Pinterest use PyCharm as their Python IDE!

It supports two versions: v2.x and v3.x.

We can run PyCharm on Windows, Linux, or Mac OS. Additionally, it contains modules and packages that help programmers develop software using Python in less time and with minimal effort. Further, it can also be customized according to the requirements of developers

Features of PyCharm:

1. Intelligent Code Editor:
2. Code Navigation:
3. Refactoring
4. Assistance for Many Other Web Technologies:
5. Support for Popular Python Web Frameworks
6. Assistance for Python Scientific Libraries

### **3.3.3 Flask**

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools

Features:

- Development server and debugger
- Integrated support for unit testing
- RESTful request dispatching
- Uses Jinja templating
- Support for secure cookies (client side sessions)
- 100% WSGI 1.0 compliant
- Unicode-based
- Extensive documentation
- Google App Engine compatibility
- Extensions available to enhance features desired

### 3.4 HARDWARE REQUIREMENTS

**Table 3.2**  
**Laptop/PC hardware specifications**

<b>S no.</b>	<b>Hardware</b>	<b>Description</b>
1	processor	Intel 3 <sup>th</sup> generation processor
2	RAM	4GB
3.	Hard Disk space	100GB

## 4. METHODOLOGY

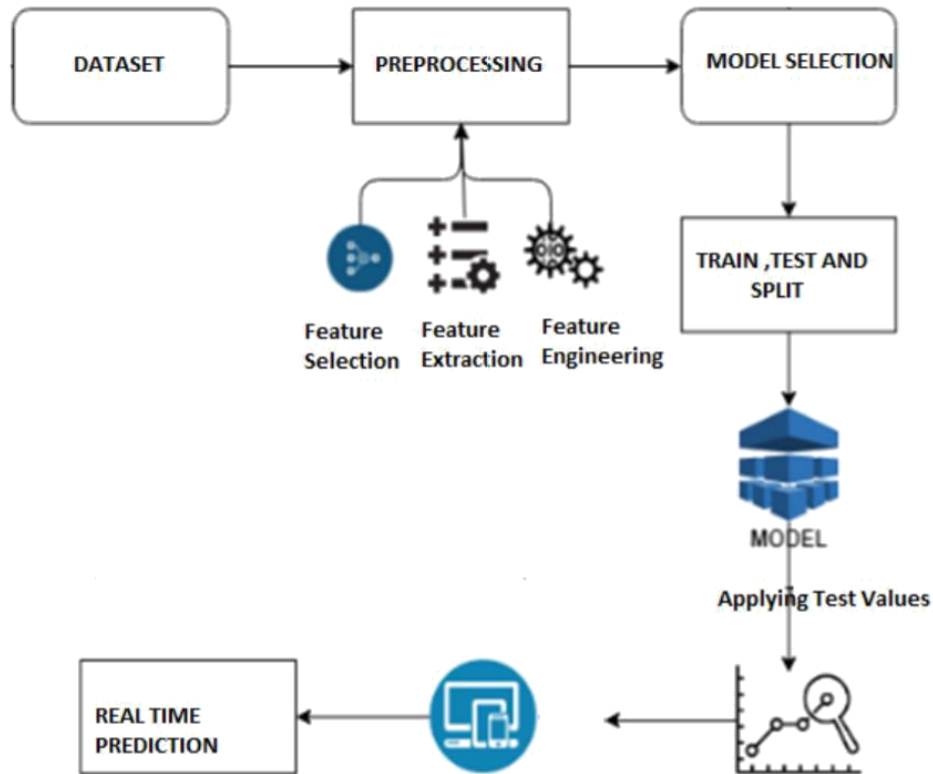
To implement the model, the standard datasets are extracted from Kaggle. The extracted datasets are based on two main characteristics: the popularity of the channel and the availability of up-to-date comments. There was no other consideration apart from these two characteristics. Therefore, the datasets were randomly selected (not based on celebrities or whatsoever). The total number of used YouTube channels is 100 and the overall samples are 10,000. Nevertheless, not all channels have presented the exact same sample share.

Before starting with the model preparation, preprocessing of the comments is done like checking whether all the characters must be in lowercase. The word which is both in uppercase and lowercase must be considered as same words and not as two different words. Then tokenization is done for each comment in the data set.

The main advantage of using the words present in the dataset is that it is capable of reducing uncertainty in the prediction of the final results as those phrases have a remarkable effect of frequency count in spam and ham comments in YouTube.

Attribute significance is a supervised characteristic that ranks attributes in a step by step manner with their significance in predicting an aim. Here Count Vectorizer is used which convert a “collection of text documents to a matrix of token counts .

## 4.1 SYSTEM ARCHITECTURE



**Fig 4.1. Sequential Flow of operations**

The datasets are trained on various models. Each model accuracy is calculated and compare the results. A model selection will be done based the performance and high accuracy. The best model is being implemented and deployed into web application.



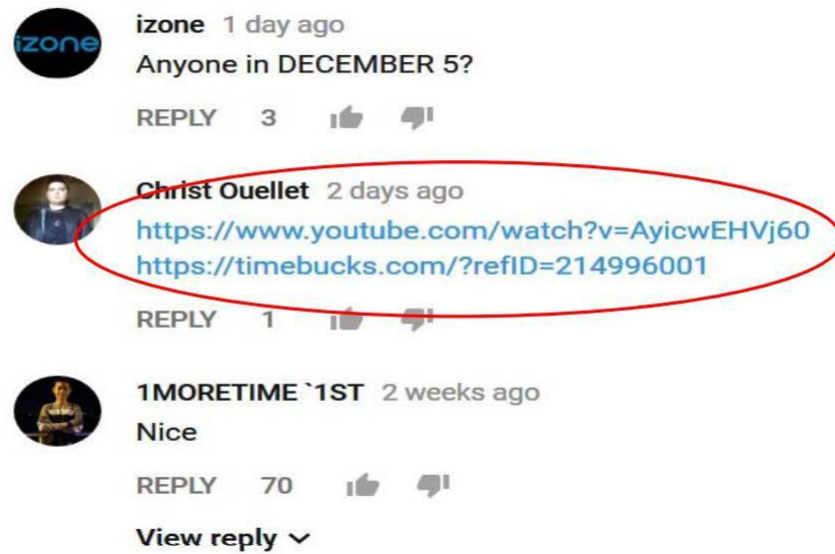


Fig 4.2 A typical example for spam comment on youtube

#### 4.1.1 Model selection

The datasets are trained on several models and model with the high accuracy is selected.

Table 4.1 Datasets used

Dataset Name	Spam	Ham	Total
Youtube-01 Psy.csv	175	175	350
Youtube-02 KatyPerry.csv	175	175	350
Youtube-03 LMFAO.csv	236	202	438
Youtube-04 Eminem.csv	245	203	448
Youtube-05 Shakira.csv	174	196	370
Spam.csv	4497	672	5169

## **A. Random forest**

Random forest is an ensemble of decision tree algorithms.

It is an extension of bootstrap aggregation (bagging) of decision trees and can be used for classification and regression problems.

In bagging, a number of decision trees are created where each tree is created from a different bootstrap sample of the training dataset. A bootstrap sample is a sample of the training dataset where a sample may appear more than once in the sample, referred to as sampling with replacement.

Bagging is an effective ensemble algorithm as each decision tree is fit on a slightly different training dataset, and in turn, has a slightly different performance. Unlike normal decision tree models, such as classification and regression trees (CART), trees used in the ensemble are unpruned, making them slightly overfit to the training dataset.

This is desirable as it helps to make each tree more different and have less correlated predictions or prediction errors.

Predictions from the trees are averaged across all decision trees resulting in better performance than any single tree in the model.

## **B. Logistic regression**

Logistic regression is used for prediction of binomial or multinomial values of a variable. It uses a statistical approach to find the outcome. The outcome is binary in nature. It uses a logit function for the prediction of probability of occurrence of binary outcome, it follows bernouli distribution, so the outcome here will be accurate either x or y. Here it works on dataset and predicts x or y that is spam or ham. Logistic regression is named for the function used at the core of the method, the logistic function.

The logistic function, also called the sigmoid function was developed by statisticians to describe properties of population growth in ecology, rising quickly and maxing out at the carrying capacity of the environment. It's an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits.

$$1 / (1 + e^{-\text{value}})$$

Where  $e$  is the base of the natural logarithms (Euler's number or the  $\text{EXP}()$  function in your spreadsheet) and value is the actual numerical value that you want to transform. Below is a plot of the numbers between -5 and 5 transformed into the range 0 and 1 using the logistic function.

### **Representation Used for Logistic Regression**

Logistic regression uses an equation as the representation, very much like linear regression. Input values ( $x$ ) are combined linearly using weights or coefficient values (referred to as the Greek capital letter Beta) to predict an output value ( $y$ ). A key difference from linear regression is that the output value being modeled is a binary values (0 or 1) rather than a numeric value.

Below is an example logistic regression equation:

$$y = e^{(b_0 + b_1 * x)} / (1 + e^{(b_0 + b_1 * x)})$$

Where  $y$  is the predicted output,  $b_0$  is the bias or intercept term and  $b_1$  is the coefficient for the single input value ( $x$ ). Each column in your input data has an associated  $b$  coefficient (a constant real value) that must be learned from your training data.

The actual representation of the model that you would store in memory or in a file are the coefficients in the equation (the beta value or  $b$ 's).

### **C. Naïve Bayes**

Naive Bayes is a classification algorithm for binary (two-class) and multi-class classification problems. The technique is easiest to understand when described using binary or categorical input values.

It is called naive Bayes or idiot Bayes because the calculation of the probabilities for each hypothesis are simplified to make their calculation tractable. Rather than attempting to calculate the values of each attribute value  $P(d_1, d_2, d_3|h)$ , they are assumed to be conditionally independent given the target value and calculated as  $P(d_1|h) * P(d_2|h)$  and so on.

This is a very strong assumption that is most unlikely in real data, i.e. that the attributes do not interact. Nevertheless, the approach performs surprisingly well on data where this assumption does not hold.

## Representation Used By Naive Bayes Models

The representation for naive Bayes is probabilities.

A list of probabilities are stored to file for a learned naive Bayes model. This includes:

- Class Probabilities: The probabilities of each class in the training dataset.
- Conditional Probabilities: The conditional probabilities of each input value given each class value.

## Multinomial Bayes

Multinomial Naive Bayes algorithm is a probabilistic learning method that is mostly used in Natural Language Processing (NLP). The algorithm is based on the Bayes theorem and predicts the tag of a text such as a piece of email or newspaper article. It calculates the probability of each tag for a given sample and then gives the tag with the highest probability as output.

Naive Bayes classifier is a collection of many algorithms where all the algorithms share one common principle, and that is each feature being classified is not related to any other feature. The presence or absence of a feature does not affect the presence or absence of the other feature. Naive Bayes is a powerful algorithm that is used for text data analysis and with problems with multiple classes. To understand Naive Bayes theorem's working, it is important to understand the Bayes theorem concept first as it is based on the latter.

Bayes theorem, formulated by Thomas Bayes, calculates the probability of an event occurring based on the prior knowledge of conditions related to an event. It is based on the following formula:

$$P(A|B) = P(A) * P(B|A)/P(B)$$

Where we are calculating the probability of class A when predictor B is already provided.

$P(B)$  = prior probability of B

$P(A)$  = prior probability of class A

$P(B|A)$  = occurrence of predictor B given class A probability.

#### **D. Support Vector Machine(SVM)**

Support Vector Machine (SVM) is a relatively simple Supervised Machine Learning Algorithm used for classification and/or regression. It is more preferred for classification but is sometimes very useful for regression as well. Basically, SVM finds a hyper-plane that creates a boundary between the types of data. In 2-dimensional space, this hyper-plane is nothing but a line.

In SVM, we plot each data item in the dataset in an N-dimensional space, where N is the number of features/attributes in the data. Next, find the optimal hyperplane to separate the data. So by this, you must have understood that inherently, SVM can only perform binary classification (i.e., choose between two classes). However, there are various techniques to use for multi-class problems. Support Vectors. Support vectors are the data points, which are closest to the hyperplane. These points will define the separating line better by calculating margins. These points are more relevant to the construction of the classifier.

**Hyperplane:** A hyperplane is a decision plane which separates between a set of objects having different class memberships.

**Margin:** A margin is a gap between the two lines on the closest class points. This is calculated as the perpendicular distance from the line to support vectors or closest points. If the margin is larger in between the classes, then it is considered a good margin, a smaller margin is a bad margin.

## 5. IMPLEMENTATION

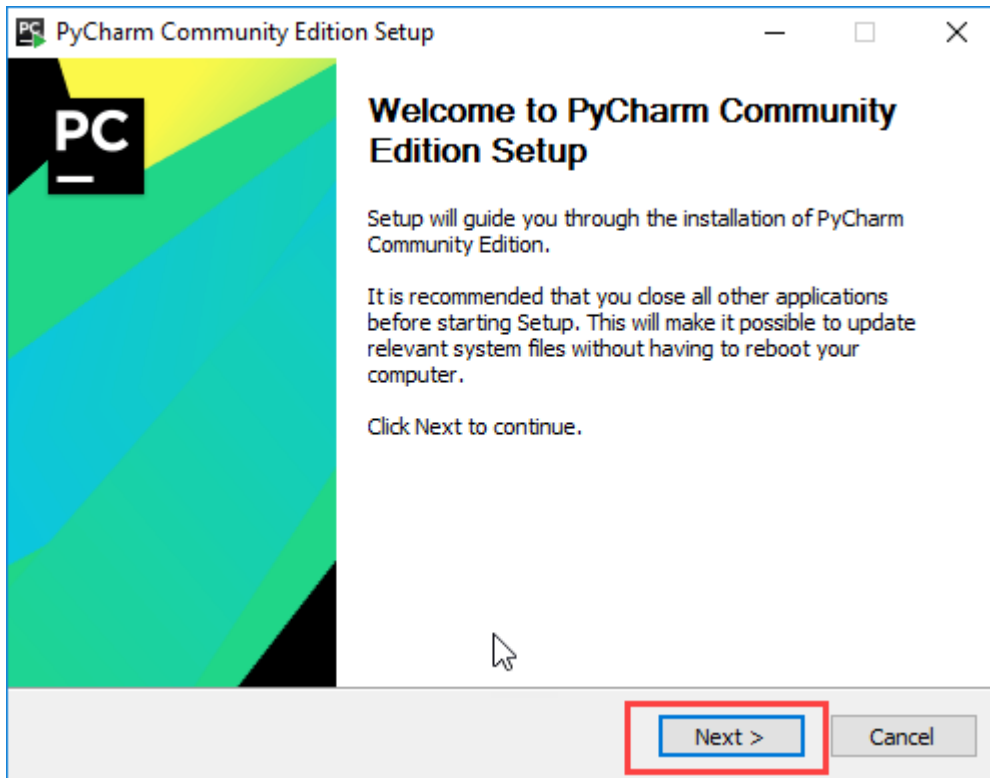
In order to run the project pycharm has to be installed. It can be installed from JetBrains' official website. PyCharm is a cross-platform editor developed by JetBrains. PyCharm provides all the tools you need for productive Python development. Below is a step by step process on how to download and install PyCharm on Windows:

First, user needs to install Python from official website as shown in the figure



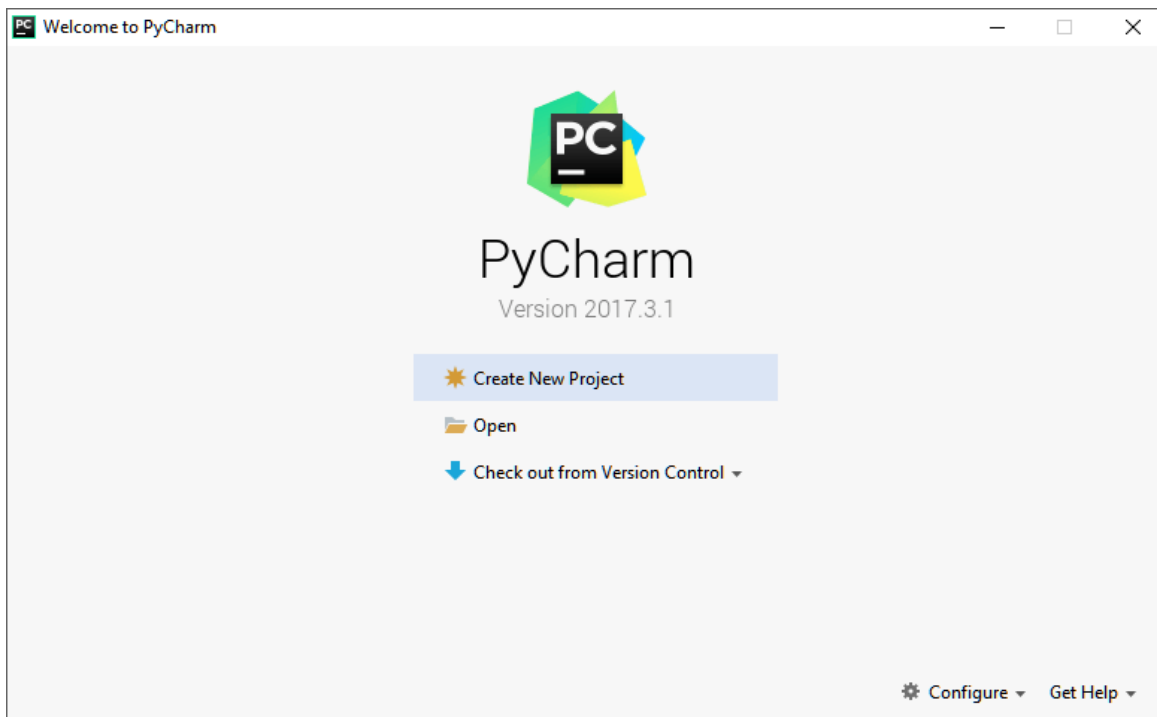
**Figure 5.1: Python installation page**

To download PyCharm visit the website <https://www.jetbrains.com/pycharm/download/> and Click the "DOWNLOAD" link under the Community Section.



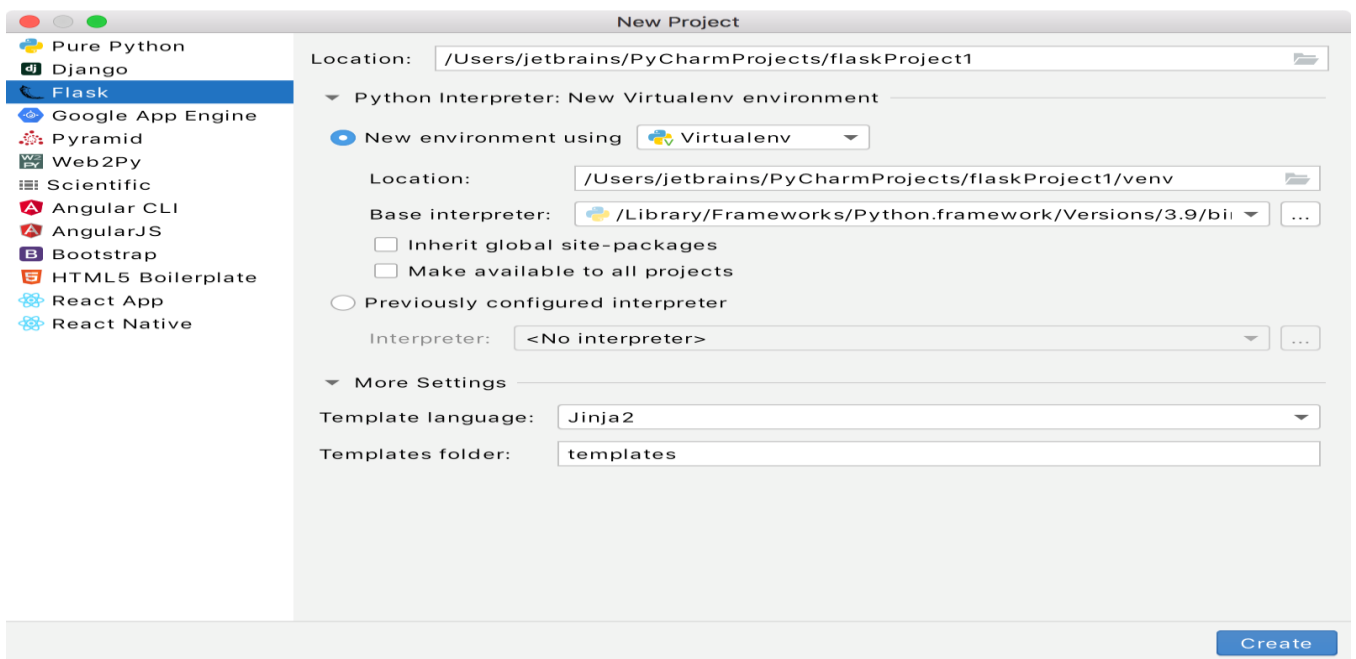
**Figure 5.2: Pycharm setup page**

- On the next screen, Change the installation path if required. Click “Next”.
- On the next screen, you can create a desktop shortcut if you want and click on “Next”.
- Choose the start menu folder. Keep selected JetBrains and click on “Install”.
- Wait for the installation to finish.
- Once installation finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the “Run PyCharm Community Edition” box first and click “Finish”.
- After you click on "Finish," the Following screen will appear.



**Figure 5.3: Pycharm welcome page**

From the main menu, choose File | New Project..., or click the New Project button in the Welcome screen. New Project dialog opens.



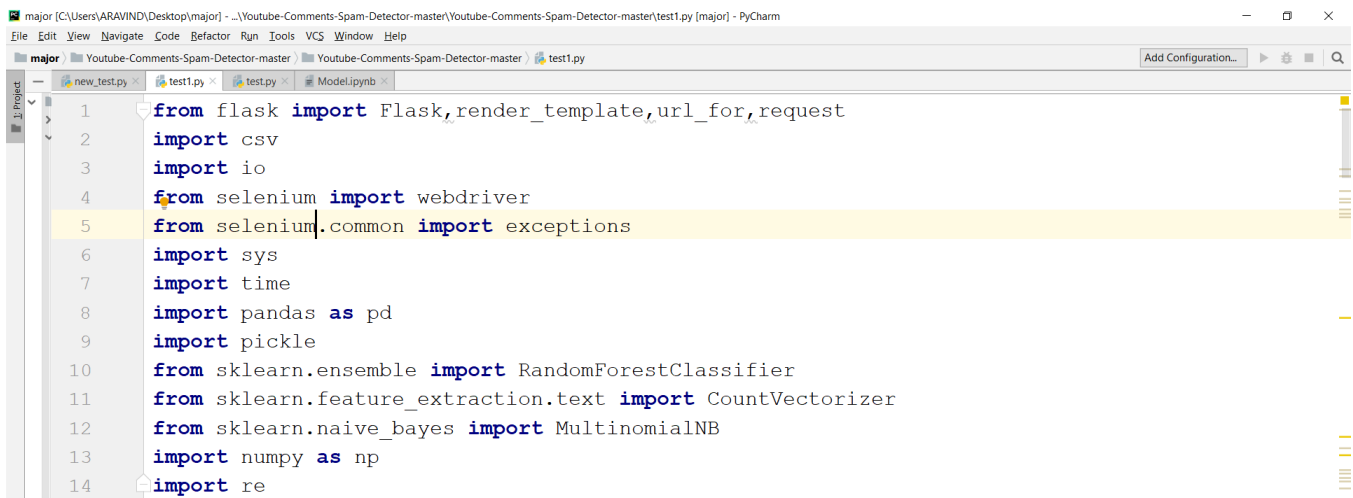
**Figure 5.4: Pycharm project create page**



The image shows the 'Run' console in PyCharm. The title bar says 'Run: Flask (app.py)'. The console output shows the following: 

```
FLASK_APP = app.py
FLASK_ENV = development
FLASK_DEBUG = 0
In folder /Users/jetbrains/PyCharmProjects/flaskProject
/Users/jetbrains/PyCharmProjects/flaskProject/venv/bin/python -m flask run
* Serving Flask app "app.py"
* Environment: development
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

**Figure 5.5 FLASK application running on local host**

The image shows a PyCharm code editor window with the file 'test1.py' open. The code contains the following imports: 

```
1 from flask import Flask, render_template, url_for, request
2 import csv
3 import io
4 from selenium import webdriver
5 from selenium.common import exceptions
6 import sys
7 import time
8 import pandas as pd
9 import pickle
10 from sklearn.ensemble import RandomForestClassifier
11 from sklearn.feature_extraction.text import CountVectorizer
12 from sklearn.naive_bayes import MultinomialNB
13 import numpy as np
14 import re
```

**Figure 5.6 Essential packages for project**

## 5.1 PACKAGES USED

These are the necessary packages to Machine learning models

### 5.1.1 Selenium package

Selenium is an open-source automation testing tool which is used for automating tests carried out on different web-browsers. Selenium is basically used to automate the testing across various web browsers. It supports various browsers like Chrome, Mozilla, Firefox, Safari, and IE, and you can very easily automate browser testing across these browsers using Selenium WebDriver. User can see live automated tests being performed on your computer screen.

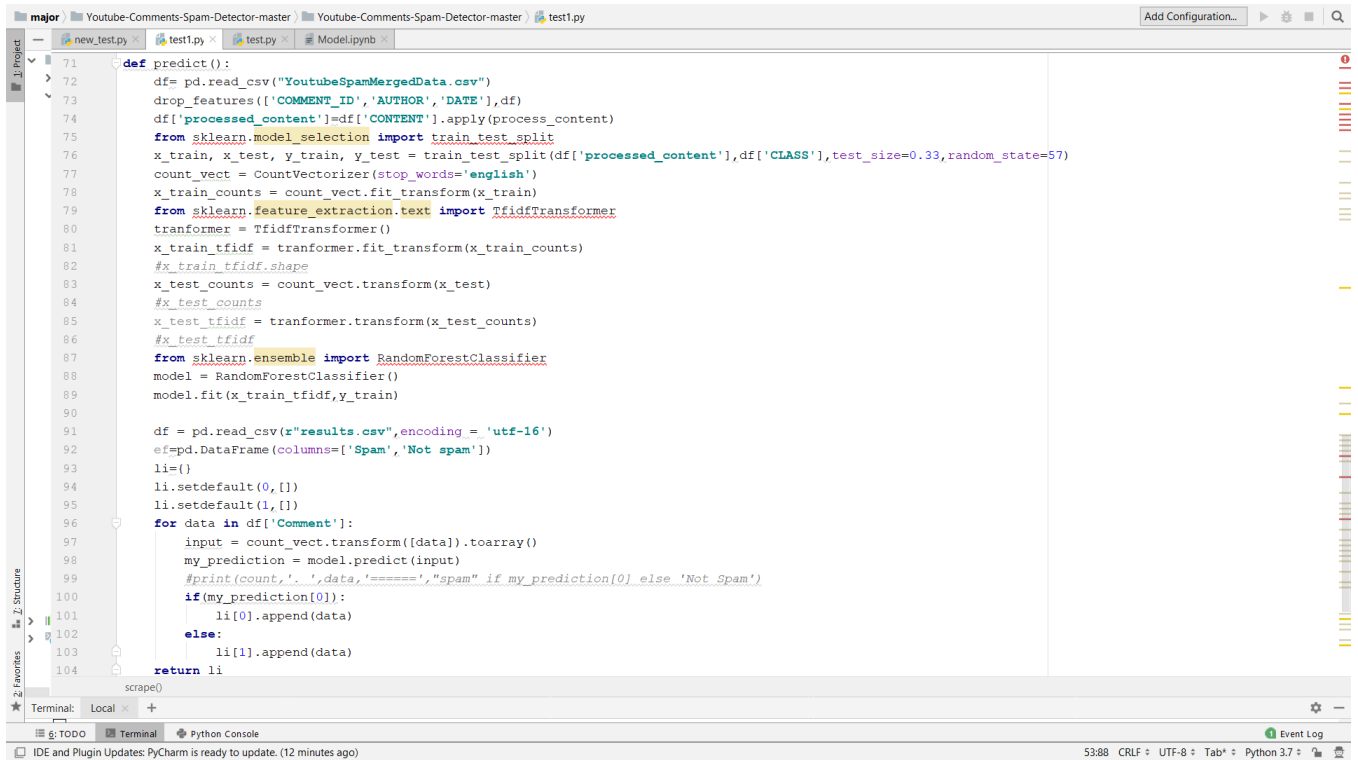
### 5.1.2 Pandas package

Pandas is an open source Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named Numpy, which provides support for multi-dimensional arrays.



Figure 5.7 Web scraping of youtube comments

- Download and replace argument with path to the driver executable.
- Download and replace argument with path to the driver executable.
- Navigates to the URL, maximizes the current window, and then suspends execution for (at least) 5 seconds (this gives time for the page to load).
- Extract the elements storing the video title and comment section.
- Youtube may have changed their HTML layouts for videos, so raise an error for sanity sake in case the elements provided cannot be found anymore.
- Scroll into view the comment section, then allow some time for everything to be loaded as necessary.
- Scroll all the way down to the bottom in order to get all the elements loaded (since Youtube dynamically loads them).
- Wait to load everything thus far.
- Calculate new scroll height and compare with last scroll height. One last scroll just in case.

The image shows a PyCharm IDE window with a project named 'major' and a sub-project 'Youtube-Comments-Spam-Detector-master'. The main editor displays a file named 'test1.py' with Python code. The code defines a 'predict()' function that reads a CSV file 'YoutubeSpamMergedData.csv', processes the data, and uses a Random Forest Classifier to predict spam or not spam. The code includes imports for pandas, sklearn, and other necessary libraries. The bottom of the IDE shows a terminal and a Python console, both of which are currently empty. The status bar at the bottom indicates the IDE and plugin updates, the current file encoding (UTF-8), and the Python version (3.7).

```
71 def predict():
72     df= pd.read_csv("YoutubeSpamMergedData.csv")
73     drop_features(['COMMENT_ID','AUTHOR','DATE'],df)
74     df['processed_content']=df['CONTENT'].apply(process_content)
75     from sklearn.model_selection import train_test_split
76     x_train, x_test, y_train, y_test = train_test_split(df['processed_content'],df['CLASS'],test_size=0.33,random_state=57)
77     count_vect = CountVecorizer(stop_words='english')
78     x_train_counts = count_vect.fit_transform(x_train)
79     from sklearn.feature_extraction.text import TfidfTransformer
80     transformer = TfidfTransformer()
81     x_train_tfidf = transformer.fit_transform(x_train_counts)
82     #x_train_tfidf.shape
83     x_test_counts = count_vect.transform(x_test)
84     #x_test_counts
85     x_test_tfidf = transformer.transform(x_test_counts)
86     #x_test_tfidf
87     from sklearn.ensemble import RandomForestClassifier
88     model = RandomForestClassifier()
89     model.fit(x_train_tfidf,y_train)
90
91     df = pd.read_csv(r"results.csv",encoding = 'utf-16')
92     ef=pd.DataFrame(columns=['Spam','Not spam'])
93     li=[]
94     li.setdefault(0,[])
95     li.setdefault(1,[])
96     for data in df['Comment']:
97         input = count_vect.transform([data]).toarray()
98         my_prediction = model.predict(input)
99         #print(count, '. ',data, '=====', "spam" if my_prediction[0] else 'Not Spam')
100         if(my_prediction[0]):
101             li[0].append(data)
102         else:
103             li[1].append(data)
104     return li
```

**Figure 5.8 Prediction and classification of comments**

- predict() function is the main implementation in this project.
- Preprocessing of data set is being performed then dataset is divided into 33% testing set and 67% training set.
- Randomforest classifier model is taken because it gave high results among the four models.

```
[ ] from sklearn.linear_model import LogisticRegression
    model = LogisticRegression()
    model.fit(x_train_tfidf,y_train)

LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                    intercept_scaling=1, l1_ratio=None, max_iter=100,
                    multi_class='auto', n_jobs=None, penalty='l2',
                    random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                    warm_start=False)
```

**Figure 5.9 Logistic regression for comparative analysis**

```
from sklearn.naive_bayes import MultinomialNB
model = MultinomialNB()
model.fit(x_train_tfidf,y_train)

MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
```

**Figure 5.10 Naive for comparative analysis**

```
[ ] from sklearn.svm import SVC, LinearSVC
    from sklearn.linear_model import SGDClassifier
    from sklearn.preprocessing import StandardScaler

[ ]
    model = SVC()
    model.fit(x_train_tfidf,y_train)

SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

**Figure 5.11 SVM for comparative analysis**

```
[ ] from sklearn.ensemble import RandomForestClassifier
    model = RandomForestClassifier()
    model.fit(x_train_tfidf,y_train)
```

```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                        criterion='gini', max_depth=None, max_features='auto',
                        max_leaf_nodes=None, max_samples=None,
                        min_impurity_decrease=0.0, min_impurity_split=None,
                        min_samples_leaf=1, min_samples_split=2,
                        min_weight_fraction_leaf=0.0, n_estimators=100,
                        n_jobs=None, oob_score=False, random_state=None,
                        verbose=0, warm_start=False)
```

**Figure 5.12 Random forest classifier for comparative analysis**

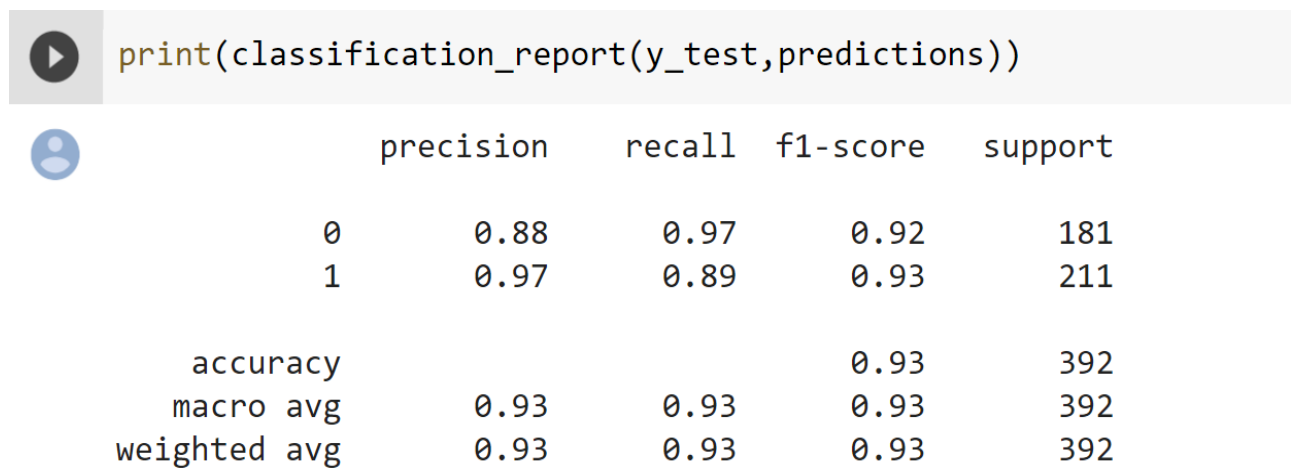
The above figures 5.9,5.10,5.11,5.12 depicts the Logistic regression, Naïve bayes, SVM, Random forest models respectively.

## 6. TESTING AND RESULTS

Test results are the outcome of the whole process of software testing life cycle. The results thus produced, offer an insight into the deliverables of a software project, significant in representing the status of the project.

```
[ ] from sklearn.model_selection import train_test_split
    x_train, x_test, y_train, y_test = train_test_split(train_data['processed_content'],train_data['CLASS'],test_size=0.2,random_state=57)
```

**Figure 6.1 Splitting the whole data into train and test sets**



**Figure 6.2 Classification report for Logistic regression**

The figure 6.2 depicts classification report for logistic regression and the accuracy calculated for 0.2 test size is 93%

```
[ ] print(classification_report(y_test,predictions))
```

	precision	recall	f1-score	support
0	0.91	0.85	0.88	181
1	0.88	0.92	0.90	211
accuracy			0.89	392
macro avg	0.89	0.89	0.89	392
weighted avg	0.89	0.89	0.89	392

**Figure 6.3 Classification report for Naïve Bayes**

The figure 6.3 depicts classification report for Naïve Bayes and the accuracy calculated for 0.2 test size is 89%

```
[ ] print(classification_report(y_test,predictions))
```

	precision	recall	f1-score	support
0	0.90	0.97	0.93	181
1	0.97	0.91	0.94	211
accuracy			0.94	392
macro avg	0.94	0.94	0.94	392
weighted avg	0.94	0.94	0.94	392

**Figure 6.4 Classification report for Random forest classifier**

The figure 6.4 depicts classification report of Random Forest and the accuracy calculated for 0.2 test size is 94%

```
print(classification_report(y_test,predictions))
```

	precision	recall	f1-score	support
0	0.89	0.97	0.93	181
1	0.97	0.90	0.93	211
accuracy			0.93	392
macro avg	0.93	0.93	0.93	392
weighted avg	0.93	0.93	0.93	392

**Figure 6.5 Classification report for SVM**

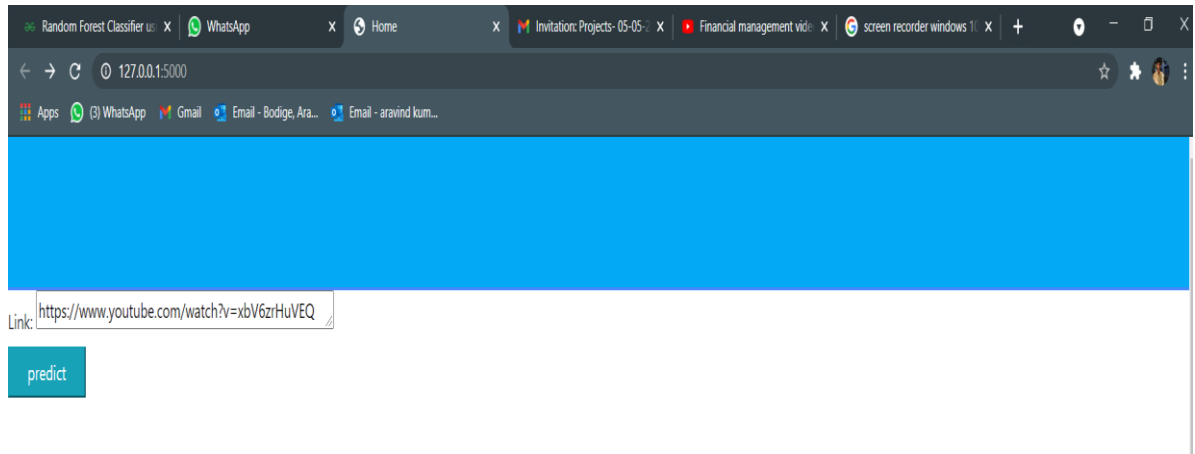
The figure 6.5 depicts classification report of SVM and the accuracy calculated for 0.2 test size is 93%.

**Table 6.1 Comparison of Accuracy for various models with different proportions of test data**

Test data size Proportion	Logistic Regression	Random Forest	Multinomial	SVM
0.2	0.93	0.93	0.89	0.93
0.33	0.93	0.95	0.89	0.93
0.43	0.94	0.95	0.87	0.94

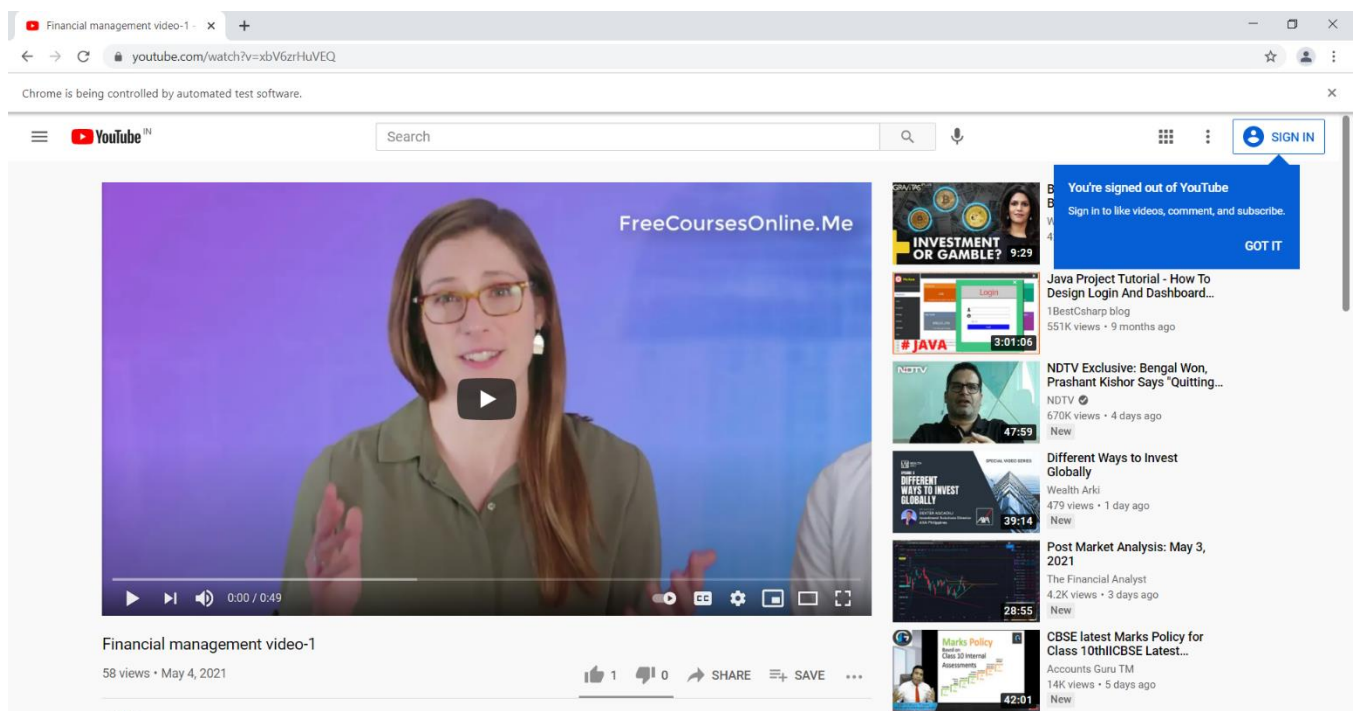
These are the various accuracies for different models at different test size proportions. Out of all the models random forest got higher accuracy that is 95%





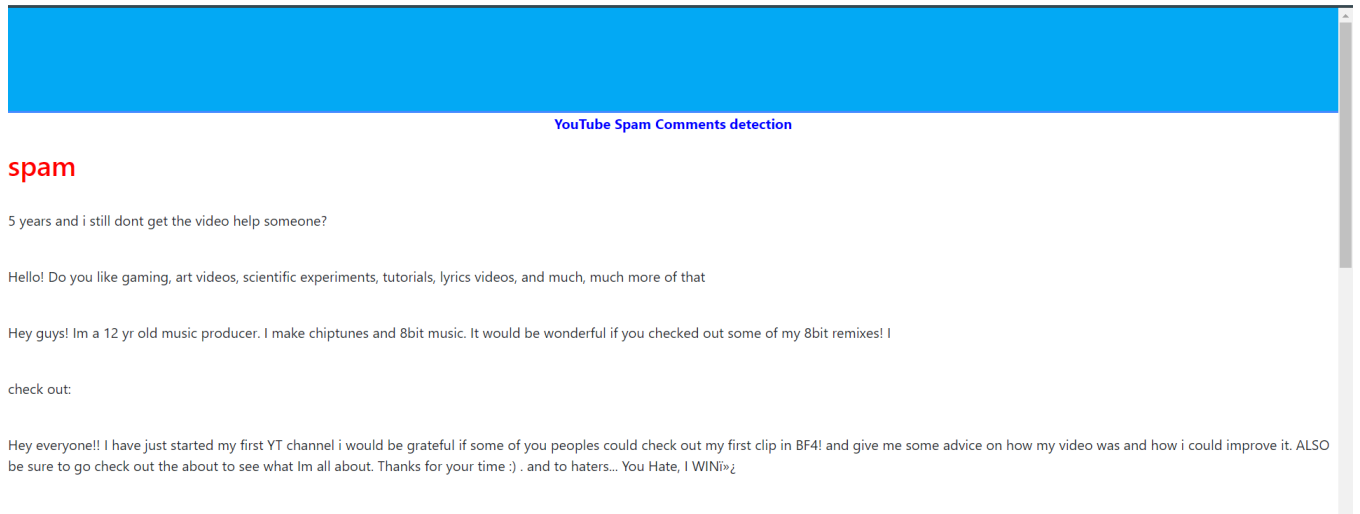
**Figure 6.6 First page on local host**

This is the first page after execution and it was hosted on localhost and here youtube video url will be provided by the user.



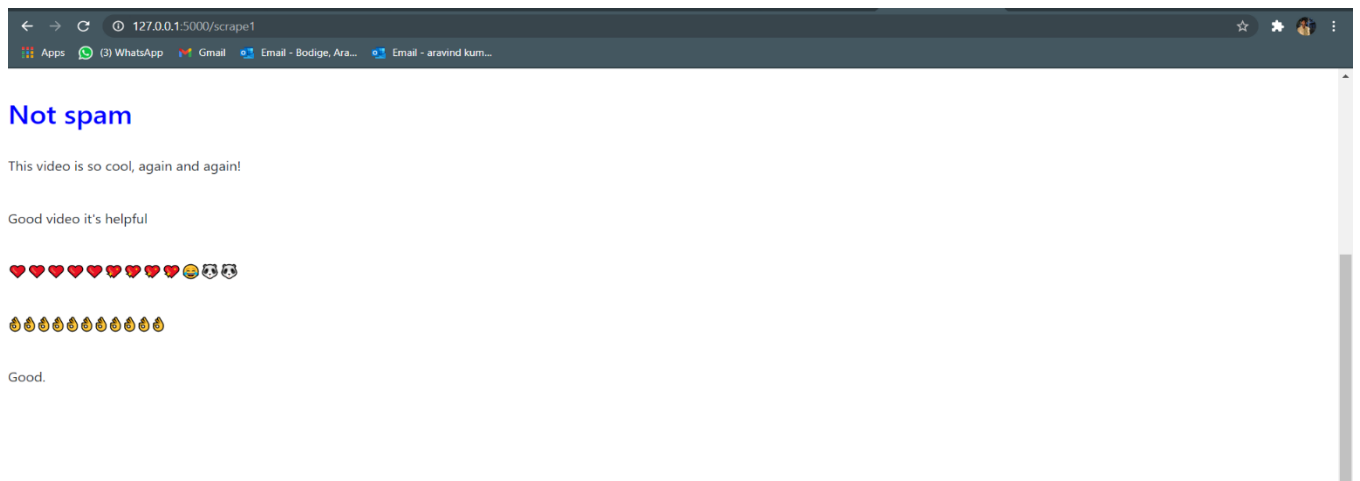
**Figure 6.7 Comments scraping on other chrome browser**

After youtube video url is given web scraping of comments are being done by using chrome driver and these comments are stored in results.csv



**Figure 6.8 Spam comments**

Figure 6.8 depicts the spam comments which are stored in results.csv which are predicted by random forest model .



**Figure 6.9 Ham comments**

Figure 6.9 depicts the Ham comments which are stored in results.csv which are predicted by random forest model .

## **7 CONCLUSION AND FUTURE SCOPE**

YouTube a social networking feature website providing one of the largest video content publication. This project used four machine learning models and tested accuracy with different test size proportions. Out of all the models, the Random classifier has given good accuracy that is 95% for the standard datasets. Unlike other existing projects, this project has the advantage of taking youtube video url and able to classify the spam and ham comments in real time.

One drawback of this project is when the comments are very high for a youtube video then it takes more time and sometimes the machine may struck.

In future the model can be modified so that more accurate results can be obtained in low processing time. Also the size of datasets can be increased for better results.

## **BIBLIOGRAPHY**

1. A Novel Approach for Youtube Video Spam Detection using Markov Decision Process, Simran Kanodia, Rachna Sasheendran, Vinod Pathari-IEE,2018
2. N-Gram Assisted Youtube Spam Comment Detection, Shreyas Aiyar, Nisha P Shetty-IEEE,2018
3. A Comparative Analysis of Common YouTube Comment Spam Filtering Techniques , Abdullah O. ,Abdulkadir Sengur,Murat Karabatak, Mashhood A. Ali- IEE,2018
4. Youtube ,Youtube moderate review & comments  
<https://support.google.com/youtube/answer/111870?hl=en>.