A

Project Report

On

**Significant Permission Identification for Machine-Learning-Based Android Malware Detection**

Submitted in partial fulfillment of the requirement for the award of Degree

Bachelor of Technology

in

computer science & engineering

Submitted by

**ALURU PRANEETH (157R1A05C1)**

**ARCHANA BANNOTH (157R1A05C7)**

**KUTIKUPPALA KRANTHI (157R1A05F1)**

Under the Guidance of

**A.RADHIKA**

(Assistant Professor)

****

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**CMR TECHNICAL CAMPUS**

(Accredited by NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi)

Recognized Under Section 2(f) & 12(B) of the UGC Act.1956,

Kandlakoya (V), Medchal Road, Hyderabad-501401.

2015-2019

**Department of Computer Science AND Engineering**



**CERTIFICATE**

This is to certify that the project report entitled **“GENRE CLASSIFICATION WITH MOVIES DATA”** is being submitted by **ALURU PRANEETH, ARCHANA BANNOTH, KUTIKUPPALA KRANTHI** bearing the **157R1A05C1, 157R1A05C7, 157R1A05F1** roll numbers in partial fulfillment of the requirements for the award of the Degree of **Bachelor of Technology** in **Computer Science & Engineering**, to the **CMR Technical Campus**, Kandlakoya, during year 2018-2019. It is certified that she has completed the project satisfactorily.

INTERNAL GUIDE

**Mrs. A. Radhika**

Assistant. Professor,

Dept of CSE,

DIRECTOR

Dr. A. Raji Reddy

HOD

**Dr. K. Srujan Raju**

**EXTERNAL EXAMINER**

**Submitted for viva voce Examination held on\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**INTERNSHIP LETTER FROM ORGANIZATION AND CERTIFICATE**

**INTERNSHIP LETTER FROM ORGANIZATION AND CERTIFICATE**

**ACKNOWLEDGEMENT**

We would like to express our sincere gratitude to our advisor, **Mrs. A. Radhika**, Assistant. Professor, CSE Dept., whose knowledge and guidance has motivated us to achieve goals we never thought possible. The time we have spent working under her supervision has truly been a pleasure.

The experience from this kind of work is great and will be useful to us in future. We thank **Dr. K. Srujan Raju**, Associate Professor & HOD, CSE Department, for encouraging us to do such research work and for providing all the facilities to carry out this project work.

It is a great pleasure to convey our thanks to our Director **Dr. A. Raji Reddy** for being cooperative throughout the course of this project. We would like to express our sincere gratitude to our Chairman Sri. **Ch. Gopal Reddy** CMR Technical Campus and the College Management for permitting us to undertake this project and providing excellent facilities to carry out our project work.

We thank all the **Faculty** members of the Department of Computer Science & Engineering for sharing their valuable knowledge with us. We extend our thanks to the **Technical Staff** of the department for their valuable suggestion to technical problems.

**DECLARATION**

We hereby declare that the work which is being presented in this dissertation entitled, “**GENRE CLASSIFICATION WITH MOVIES DATA**”, submitted towards the partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering**, **CMR Technical Campus**, Kandlakoya is an authentic record of our own work carried out under the supervision of **Mrs. A. Radhika, Assistant Professor, Department of CSE,** CMR Technical Campus, Kandlakoya.

To the best of our knowledge and belief, this project bears no resemblance with any report submitted to JNTUH or any other University for the award of any degree or diploma.

**ALURU PRANEETH (157R1A05C1)**

**ARCHANA BANNOTH (157R1A05C7)**

**KUTIKUPPALA KRANTHI (157R1A05F1)**

Date: 08TH OCTOBER 2018

Place: HYDERABAD

**ABSTRACT**

Malware has always been a problem in regards to any technological advances in the software world. Thus, it is to be expected that smart phones and other mobile devices are facing the same issues. In this paper, a practical and effective anomaly based malware detection framework is proposed with an emphasis on Android mobile computing platform. A dataset consisting of both benign and malicious applications (apps) were installed on an Android device to analyze the behavioural patterns. We first generate the system metrics (feature vector) from each app by executing it in a controlled environment. Then, a variety of machine learning algorithms: Decision Tree, K Nearest Neighbour, Logistic Regression, Multilayer Perceptron Neural Network, Naive Bayes, Random Forest, and Support Vector Machine are used to classify the app as benign or malware.

Each algorithm is assessed using various performance criteria to identify which ones are more suitable to detect malicious software. The results suggest that Random Forest and Support Vector Machine provide the best outcomes thus making them the most effective techniques for malware detection.

malware detection.

**CONTENTS**

**PAGE NO.**

[LIST OF FIGURES vii](#_Toc423989484)i

[LIST OF ABBREVIATIONS](#_Toc423989486) ix

[CHAPTER 1 INTRODUCTION 1](#_Toc423989487)

[1.1 MOTIVATION](#_Toc423989488) 2

[1.2 EXISTING SYSTEM](#_Toc423989489) 2

[1.3 Problem statement](#_Toc423989490) 3

[1.4 Proposed system](#_Toc423989491) 3

[1.5 Objective](#_Toc423989492) 3

[CHAPTER 2 REVIEW OF LITERATURE 4](#_Toc423989493)

[2.1 REFERENCE PAPER-I 4](#_Toc423989494)

[2.2 REFERENCE PAPER-II 4](#_Toc423989495)

[2.3 REFERENCE PAPER-III 4](#_Toc423989496)

[2.4 Requirement specifications 5](#_Toc423989500)

[2.4.1 HardwareRequirements 5](#_Toc423989501)

[2.4.2 Software Requirements 5](#_Toc423989502)

[2.5 Domain Information 5](#_Toc423989503)

[CHAPTER 3 EXPERIMENTATION ANALYSIS 6](#_Toc423989504)

3.1 [ARCHITECTURE OF PROPOSED SYSTEM](#_Toc423989509) 6

3.2 UML DIAGRAMS 9

3.3 WHAT IS THE USE OF UML 9

3.4 PACKAGES USED IN PROGRAM 19

3.5 SAMPLE CODE 30

## [CHAPTER 4 RESULTS AND DISCUSSION](#_Toc423989516) 35

CHAPTER 5 CONCLUSION AND FUTURE SCOPE 38

5.1 CONCLUSION 38

5.2 FUTURE SCOPE 38

**REFERENCES 39**

# LIST OF FIGURES

**PAGE NO.**

[Figure 4.1.1 Use case Diagram](#_Toc423989594) 6

[Figure 4.1.2 Class Diagram](#_Toc423989595) 7

[Figure 4.1.3 Sequence Diagram](#_Toc423989596) 8

Figure 5.1 Screen-1 14 [Figure 5.2 Screen-2](#_Toc423989601) 14

Figure 5.3 Screen-3 15

# LIST OF ABBREVIATIONS

SVM : SUPPORT VECTOR MACHINE

Support Vector Machine” (SVM) is a supervised machine learningalgorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems.Support Vectors are simply the co-ordinates of individual observation.

KNN: K- NEAREST NEIGHBOURS

*k*-nearest neighbors algorithm (*k*-NN) is a [non-parametric](https://en.wikipedia.org/wiki/Non-parametric_statistics) method used for [classification](https://en.wikipedia.org/wiki/Statistical_classification) and [regression](https://en.wikipedia.org/wiki/Regression_analysis). *k*-NN is a type of [instance-based learning](https://en.wikipedia.org/wiki/Instance-based_learning), or [lazy learning](https://en.wikipedia.org/wiki/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](https://en.wikipedia.org/wiki/Machine_learning) algorithms.

# chapter 1

# introduction

Malware, short for malicious software, is a general term used to refer to a variety of forms of hostile or intrusive software such as viruses, worms, spyware, Trojan horses,rootkits, and backdoors. A common feature of Malware is that they are specifically designed to damage, disrupt, steal,or in general inflict some other bad or illegitimate actions. Malware can literally infect any computing machines running user programs (or applications), and the propagation and prevention of the malware have been well studied for personal computers. But for smartphone devices, our solutions for finding malware in the mobile platform are far behind the pace of the increasing popularity of the mobile applications. A recent report has shown that there are about 700,000 Android Apps currently available on the market. This popularity of the Android system has led to a huge increase in the spreading of Android malware. These malware are mainly distributed in markets operated by third parties, but even the Google Android Market cannot guarantee that all of its listed applications are threat free. The threats for Android include Phishing, Banking-Trojans, Spyware, Bots, Root Exploits, SMS Fraud, Premium Dialers and Fake Installers. There have also been reports about Download-Trojans Apps that download their malicious code after installation which means that these Apps cannot be easily detected by Google’s technology during publication in the Google Android Market. In summary, malware applications commonly use following three types of penetration techniques for installation, activation, and running on the Android system: Repackaging is one of the most common techniques for malware developers to install malicious applications on an Android platform. These types of approaches normally start from popular legitimate Apps and misuse them as malware. The developers normally download popular Apps, disassemble them, add their own malicious codes, and then re-assemble and upload the new App to official or alternative markets. Updating technique is more difficult for detection.

## 

## MOTIVATION

Websites like Netflix and HBO Go, provide lists of movies based on genres, this makes it easier for user to select the movie that interests him/her based on the genre he/she is more inclined towards. Tagging of movies is a complex process and usually involves a manual process where the movies are assigned to one or more genres based on the suggestions sent by the users and consumers. If we can automate this process of movie tagging, not only will it be fast , save human effort but it will be more accurate than an untrained human as well.

## EXISTING SYSTEM

For each newly collected unknown Android app, it will be ﬁrst parsed through the unzipper and decompiler to get the smali codes, then its API calls will be extracted from the smali codes, and the relationships among these API calls will be further analyzed. Basedontheextractedfeaturesandusingtheconstructed classiﬁcation model, this app will be labeled as either benign or malicious.

**Disadvantages of Existing System:**

* Efficiency levels are very low

## Problem statement

The problem is to train the system to differentiate the types of genres and to display the maximum genre percentage by comparing other genres.Version 5.1.1). As a result, the number of goodwares we used is 176 and that of malware is 59. In this proposed system, The dataset used is the same one used in Damshenasa et al. on M0droid . It contains a set of both benign and malicious software. The dataset comprises of 200 goodware and 115 malware apps, which will be installed on the mobile device. It compromises of a variety of “.apk” files such as games, widgets, and other apps. It should be noted that not all “.apk” files in the dataset will be used since some were not compatible with the operating system of the device that we use (unrooted Google/LG Nexus 4 running Android Lollipop

**Advantages:**

* Prediction levels are very high.
* Apk has predictable content. Application behaviour can be learned by simply reading the file.
* Checks are safe

## Objective

This project will be a study which determines how various modifications in the malware apps change the accuracy of genre percentage for various classifiers and techniques which allows the end users to easily visualize the Malware App based on maximum percentage of that particular malware app.

# literature SURVEY

## REFERENCE PAPER-I

# Significant permission identification for machine-learning-based android malware detection.

**Published in:**[IEEE Transactions on Industrial Informatics](https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=9424) ( Volume: 14 , [Issue: 7](https://ieeexplore.ieee.org/xpl/tocresult.jsp?isnumber=8402070) , July 2018 )

  The alarming growth rate of malicious apps has become a serious issue that sets back the prosperous mobile ecosystem. A recent report indicates that a new malicious app for Android is introduced every 10 s. To combat this serious malware campaign, we need a scalable malware detection approach that can effectively and efficiently identify malware apps. Numerous malware detection tools have been developed, including system-level and network-level approaches. However, scaling the detection for a large bundle of apps remains a challenging task. In this paper, we introduce Significant Permission IDentification (SigPID), a malware detection system based on permission usage analysis to cope with the rapid increase in the number of Android malware. Instead of extracting and analyzing all Android permissions, we develop three levels of pruning by mining the permission data to identify the most significant permissions that can be effective in distinguishing between benign and malicious apps. SigPID then utilizes machine-learning-based classification methods to classify different families of malware and benign apps.

## REFERENCE PAPER-II

## Android malware detection a survey by Raima Zachariah; Akash; Mohammed Sajmal Yousef; Anu Mary Chacko

**Published in:**[2017 IEEE International Conference on Circuits and Systems (ICCS)](https://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=8320243)

In this survey paper, we aim to briefly discuss the different techniques used in Android malware detection and highlight their advantages and limitations. The popularity of Android mobile devices has gone up in our lives and are being used for handling a lot of our personal and confidential information. Hence t hey are now an ideal target for attackers. Android based smart-phone users can download a lot of free applications from Android Application Market/Play Store. These downloaded apps some-times contain malware applications that can take possession of private information from users. Apps in queue for launching on Play Store generally undergo an automatic security test where crawl operations like typing, tapping and swiping are performed periodically and analysis results are generated. In this survey paper, we aim to briefly discuss the different techniques used in Android malware detection and highlight their advantages and limitations.

## 2.3 REFERENCE PAPER-III

# Malware detection system based on machine learning methods for android operating systems.

**Published in:** [2017 25th Signal Processing and Communications Applications Conference (SIU)](https://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7951214)

In this paper, it has been made a research on mobile malware and malware detection techniques. Within the scope of the study, a a permission based detection system based on the machine learning methods for Android malware was developed. Developed system is analyzed by using Random Forest, Support Vector Machine and Artificial Neural Networks algorithms. Mobile devices begin to spread increasingly recently to offer a lot of services which personal computers offer. This condition has led to increase in the number of security threats in mobile devices and services. In this paper, it has been made a research on mobile malware and malware detection techniques. Within the scope of the study, a a permission based detection system based on the machine learning methods for Android malware was developed. Developed system is analyzed by using Random Forest, Support Vector Machine and Artificial Neural Networks algorithms.

# Experimentation analysis

## ARCHITECTURE OF PROPOSED SYSTEM

The architecture of the proposed system consists four modules namely:

* The training phase
* The prediction phase
* The NLTK implementation and
* Classification

#### The Training Phase:

In the training phase the system gets trained with the data and preprocesses the data based upon the sentiment. This is given as input to the NLTK implementation and the system gets trained using this NLTK package.

#### The Predicting Phase:

In the predicting phase the system tests the text entered by the user and preprocesses the data using its trained dataset and then features the result to NLTK implementation.

#### NLTK Implementation:

The NLTK module is a massive tool kit, aimed at helping you with the entire Natural Language Processing (NLP) methodology. NLTK will aid you with everything from splitting sentences from paragraphs, splitting up words, recognizing the part of speech of those words, highlighting the main subjects, and then even with helping your machine to understand what the text is all about. In this series, we're going to tackle the field of opinion mining, or sentiment analysis.

In order to get started, you are going to need the NLTK module, as well as Python.If you do not have Python yet, go to [Python.org](https://www.python.org/) and download the latest version of Python if you are on Windows. If you are on Mac or Linux, you should be able to run an apt-get install python3.

Here in NLTK the package is imported and the positive and the negative words are trained by the system using this package.

#### Classification:

Classification is the set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known. It is a supervised learning approach in which the computer program learns from the data input given to it and then uses this learning to classify new observation. This data set may simply be bi-class (like identifying whether the person is male or female or that the mail is spam or non-spam) or it may be multi-class too.

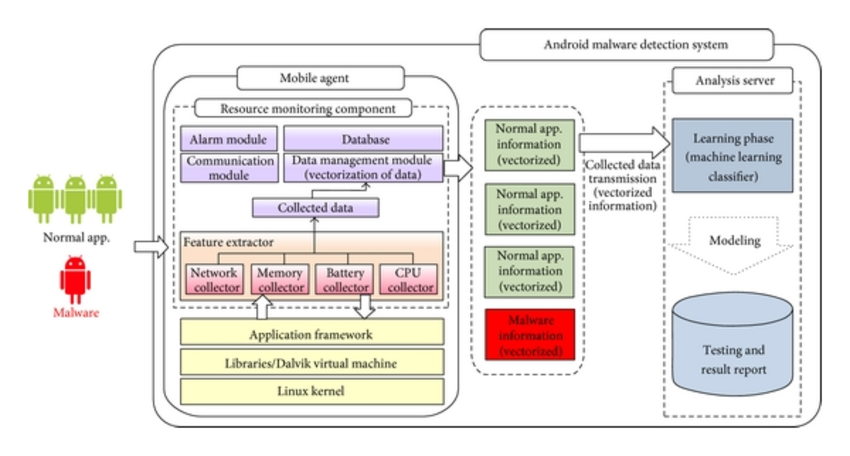


Figure 3.1 Architecture Diagram

## 4. Requirement specifications

### 4.1 Hardware Requirements

### System : Windows

### Hard Disk : 120 GB.

### Monitor : 15’’ LED

### Input Devices : Keyboard, Mouse

### Ram : 4GB

### 4.2 Software Requirements

* Python
* Included development tools: Jupyter Notebook, Spyder
* Compatible tools: PyCharm

## 4.3 Domain Information

A classification problem is when the output variable is a category, such as “red” or “blue” or “disease” and “no disease”. A classification model attempts to draw some conclusion from observed values. Given one or more inputs a classification model will try to predict the value of one or more outcomes.For example, when filtering emails “spam” or “not spam”, when looking at transaction data, “fraudulent”, or “authorized”. In short Classification either predicts categorical class labels or classifies data (construct a model) based on the training set and the values (class labels) in classifying attributes and uses it in classifying new data. There are a number of classification models. Classification models include logistic regression, decision tree, random forest, gradient-boosted tree, multilayer perceptron, one-vs-rest, and Naive Bayes.

Eg : Predicting the gender of a person Predicting whether monsoon will be normal next year.

A regression problem is when the output variable is a real or continuous value, such as “salary” or “weight”. Many different models can be used, the simplest is the linear regression. It tries to fit data with the best hyper-plane which goes through the points

**Eg:**Predicting age of a person (because it is a real value, predicting nationality is categorical, whether stock price will increase is discreet-yes/no answer, predicting whether a document is related to UFO is again discreet- a yes/no answer).

**Random Forest**

Random forests or random decision forests are an [ensemble learning](https://en.wikipedia.org/wiki/Ensemble_learning) method for [classification](https://en.wikipedia.org/wiki/Statistical_classification), [regression](https://en.wikipedia.org/wiki/Regression_analysis) and other tasks that operates by constructing a multitude of [decision trees](https://en.wikipedia.org/wiki/Decision_tree_learning) at training time and outputting the class that is the [mode](https://en.wikipedia.org/wiki/Mode_(statistics)) of the classes (classification) or mean prediction (regression) of the individual trees.[[1]](https://en.wikipedia.org/wiki/Random_forest#cite_note-ho1995-1)[[2]](https://en.wikipedia.org/wiki/Random_forest#cite_note-ho1998-2) Random decision forests correct for decision trees' habit of [overfitting](https://en.wikipedia.org/wiki/Overfitting) to their [training set](https://en.wikipedia.org/wiki/Test_set).[[3]](https://en.wikipedia.org/wiki/Random_forest#cite_note-elemstatlearn-3):587–588

The first algorithm for random decision forests was created by [Tin Kam Ho](https://en.wikipedia.org/wiki/Tin_Kam_Ho)[[1]](https://en.wikipedia.org/wiki/Random_forest#cite_note-ho1995-1) using the [random subspace method](https://en.wikipedia.org/wiki/Random_subspace_method),[[2]](https://en.wikipedia.org/wiki/Random_forest#cite_note-ho1998-2) which, in Ho's formulation, is a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg.[[4]](https://en.wikipedia.org/wiki/Random_forest#cite_note-kleinberg1990-4)[[5]](https://en.wikipedia.org/wiki/Random_forest#cite_note-kleinberg1996-5)[[6]](https://en.wikipedia.org/wiki/Random_forest#cite_note-kleinberg2000-6)

An extension of the algorithm was developed by [Leo Breiman](https://en.wikipedia.org/wiki/Leo_Breiman)[[7]](https://en.wikipedia.org/wiki/Random_forest#cite_note-breiman2001-7) and Adele Cutler,[[8]](https://en.wikipedia.org/wiki/Random_forest#cite_note-rpackage-8) who registered[[9]](https://en.wikipedia.org/wiki/Random_forest#cite_note-9) "Random Forests" as a [trademark](https://en.wikipedia.org/wiki/Trademark)(as of 2019, owned by [Minitab, Inc.](https://en.wikipedia.org/wiki/Minitab)).[[10]](https://en.wikipedia.org/wiki/Random_forest#cite_note-10) The extension combines Breiman's "[bagging](https://en.wikipedia.org/wiki/Bootstrap_aggregating)" idea and random selection of features, introduced first by Ho[[1]](https://en.wikipedia.org/wiki/Random_forest#cite_note-ho1995-1) and later independently by Amit and [Geman](https://en.wikipedia.org/wiki/Donald_Geman" \o "Donald Geman)[[11]](https://en.wikipedia.org/wiki/Random_forest#cite_note-amitgeman1997-11) in order to construct a collection of decision trees with controlled variance.

**3.1.1 MODULES:**

**User**: The user opens the software and then runs it. The user waits for the system to generate the result.

**System**: The system gets trained with the dataset of benign and malware and then takes the input from the user. The system verifies the input and process the result based on the matched attributes with the particular dataset.

**Admin**: The admin adds any new field in the dataset and makes the system help in getting trained with the dataset. He adds new attributes to the dataset if they are not present in the dataset. When the user runs the Software and if that malware is not present in the dataset the admin trains the system with that new word into the dataset and then the system displays the result. The admin also provides the security to the system and manage to rebuild the system if the system is crashed.

## UML Diagrams

The Unified Modelling Language (UML) diagrams represent the flow and structure of the process in the project.Simply put, UML is a modern approach to modeling and documenting software. In fact, it’s one of the most popular [business process modeling techniques](https://tallyfy.com/business-process-modeling-techniques). It is based on **diagrammatic representations** of software components. As the old proverb says: “a picture is worth a thousand words”. By using visual representations, we are able to better understand possible flaws or errors in software or business processes.

## What is the use of UML?

Mainly, UML has been used as a general-purpose modeling language in the field of software engineering. However, it has now found its way into the documentation of several [business processes](https://tallyfy.com/business-process) or [workflows](https://tallyfy.com/what-is-a-workflow/). For example, activity diagrams, a type of UML diagram, can be used as a replacement for flowcharts. They provide both a more standardized way of modeling workflows as well as a wider range of features to improve readability and efficacy. UML itself finds different uses in software development and business process documentation:

#### Sketch

UML diagrams, in this case, are used to communicate different aspects and characteristics of a system. However, this is only a top-level view of the system and will most probably not include all the necessary details to execute the project until the very end.

* + **Forward Design** – The design of the sketch is done before coding the application. This is done to get a better view of the system or workflow that you are trying to create. Many design issues or flaws can be revealed, thus improving the overall project health and well-being.
  + **Backward Design** – After writing the code, the UML diagrams are drawn as a form of documentation for the different activities, roles, actors, and workflows.

#### Blueprint

In such a case, the UML diagram serves as a complete design that requires solely the actual implementation of the system or software. Often, this is done by using [CASE](https://en.wikipedia.org/wiki/Computer-aided_software_engineering) tools (Computer Aided Software Engineering Tools). The main drawback of using CASE tools is that they require a certain level of expertise, user training as well as management and staff commitment.

#### Pseudo Programming Language

UML is not a stand-alone programming language like Java, C++ or Python, however, with the right tools, it can turn into a pseudo programming language. In order to achieve this, the whole system needs to be documented in different UML diagrams and, by using the right software, the diagrams can be directly translated into code. This method can only be beneficial if the time it takes to draw the diagrams would take less time than writing the actual code.

Despite UML having been created for modeling software systems, it has found several adoptions in business fields or non-software systems.

#### Types of UML Diagrams

There are several types of UML diagrams and each one of them serves a different purpose regardless of whether it is being designed before the implementation or after (as part of documentation).

The two most broad categories that encompass all other types are **Behavioral** UML diagram and **Structural** UML diagram. As the name suggests, some UML diagrams try to analyze and depict the structure of a system or process, whereas other describe the behavior of the system, its actors, and its building components. The different types are broken down as follows:

* Use Case Diagram
* Interaction Overview Diagram
* Timing Diagram
* State Machine Diagram
* Communication Diagram
* Sequence Diagram
* Behavioural Diagram
* Activity Diagram

#### Structural UML Diagram

* Class Diagram
* Object Diagram
* Component Diagram
* Composite Diagram
* Composite Structure Diagram
* Deployement Diagarm
* Package Diagram
* Profile Diagram

#### Use case Diagram

**Use case diagrams** are usually referred to as behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more **external users** of the system (actors). Use case diagrams are in fact twofold - they are both [behavior diagrams](https://www.uml-diagrams.org/uml-25-diagrams.html#behavior-diagram), because they describe behavior of the system, and they are also [structure diagrams](https://www.uml-diagrams.org/uml-25-diagrams.html#structure-diagram) - as a special case of class diagrams where classifiers are restricted to be either [actors](https://www.uml-diagrams.org/use-case-actor.html) or [use cases](https://www.uml-diagrams.org/use-case.html) related to each other with [associations](https://www.uml-diagrams.org/association.html).

* Each use case should provide some observable and valuable result to the actors or other stakeholders of the system.
* Use case diagrams give a graphic overview of the actors involved in a system, different functions needed by those actors and how these different functions interact.
* It’s a great starting point for any project discussion because you can easily identify the main actors involved and the main processes of the system.
* You can [create use case diagrams](https://creately.com/diagram-type/use-case?utm_source=umltypes&utm_medium=blog&utm_campaign=tutorialposts) using our tool and/or get started instantly using our use case templates.
* A cornerstone part of the system is the [functional requirements](https://reqtest.com/requirements-blog/functional-vs-non-functional-requirements/) that the system fulfils. Use Case diagrams are used to analyze the system’s [high-level requirements](http://www.testablerequirements.com/testablerequirements/ident_hlrs.htm).

These requirements are expressed through different use cases. We notice three main components of this UML diagram:

**Functional requirements** – represented as use cases; a verb describing an action

**Actors** – they interact with the system; an actor can be a human being, an organization or an internal or external application

**Relationships** between actors and use cases – represented using straight arrows

In use diagrams, within the circular containers, we express the actions that the actors perform. Such actions are: purchasing and paying for the stock, checking stock quality, returning the stock or distributing it. As you might have noticed, use case UML diagrams are good for showing dynamic behaviors between actors within a system, by simplifying the view of the system and not reflecting the details of implementation.

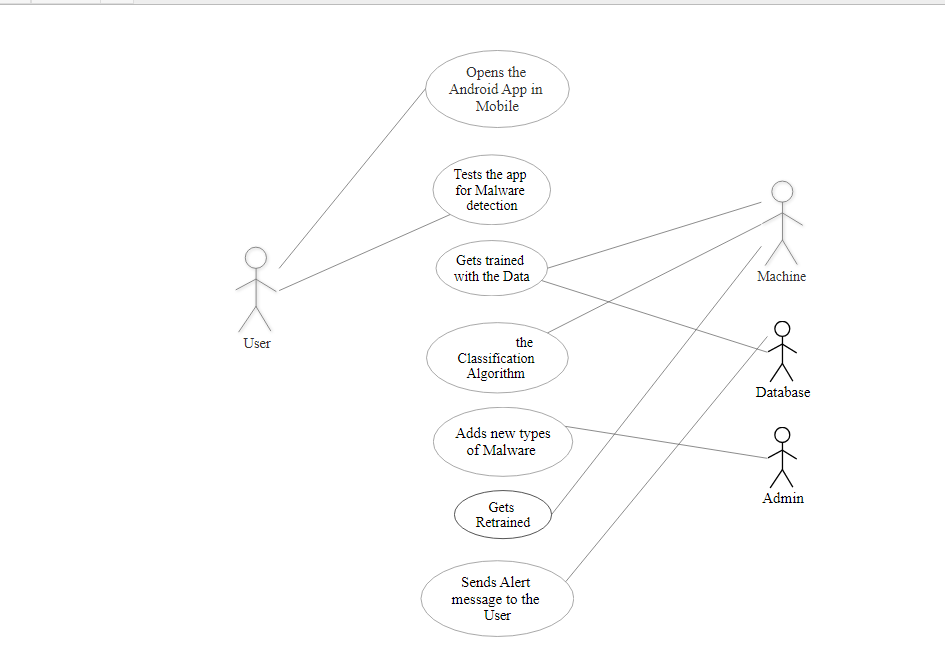


Figure 3.3.1.7 Use case Diagram

In the above Use Case Diagram, user, admin, system and database are actors and Opens the Android Mobile App, Tests the app for malware detection, add new types of malwares, Gets trained, Sends Alert message to the user, are use cases. The user's functionality is run the software, System's functionality is to get trained and display the result, Admin's functionality is to add new malware and Database functionality is to store the data.

#### ****Activity diagram****

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system.Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc

## Purpose of Activity Diagrams

The basic purposes of activity diagrams is similar to other four diagrams. It captures the dynamic behavior of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.

It does not show any message flow from one activity to another. Activity diagram is sometimes considered as the flowchart. Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched, concurrent, and single.

The purpose of an activity diagram can be described as −

* Draw the activity flow of a system.
* Describe the sequence from one activity to another.
* Describe the parallel, branched and concurrent flow of the system.

## How to Draw an Activity Diagram?

Activity diagrams are mainly used as a flowchart that consists of activities performed by the system. Activity diagrams are not exactly flowcharts as they have some additional capabilities. These additional capabilities include branching, parallel flow, swimlane, etc.

Before drawing an activity diagram, we must have a clear understanding about the elements used in activity diagram. The main element of an activity diagram is the activity itself. An activity is a function performed by the system. After identifying the activities, we need to understand how they are associated with constraints and conditions.

Before drawing an activity diagram, we should identify the following elements −

* Activities
* Association
* Conditions
* Constraints

Once the above-mentioned parameters are identified, we need to make a mental layout of the entire flow. This mental layout is then transformed into an activity diagram.

Following is an example of an activity diagram for order management system. In the diagram, four activities are identified which are associated with conditions. One important point should be clearly understood that an activity diagram cannot be exactly matched with the code. The activity diagram is made to understand the flow of activities and is mainly used by the business users

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system.

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc

## Purpose of Activity Diagrams

The basic purposes of activity diagrams is similar to other four diagrams. It captures the dynamic behavior of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.

It does not show any message flow from one activity to another. Activity diagram is sometimes considered as the flowchart. Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched, concurrent, and single.

The purpose of an activity diagram can be described as −

* Draw the activity flow of a system.
* Describe the sequence from one activity to another.
* Describe the parallel, branched and concurrent flow of the system.

## How to Draw an Activity Diagram?

Activity diagrams are mainly used as a flowchart that consists of activities performed by the system. Activity diagrams are not exactly flowcharts as they have some additional capabilities. These additional capabilities include branching, parallel flow, swimlane, etc.

Before drawing an activity diagram, we must have a clear understanding about the elements used in activity diagram. The main element of an activity diagram is the activity itself. An activity is a function performed by the system. After identifying the activities, we need to understand how they are associated with constraints and conditions.

Before drawing an activity diagram, we should identify the following elements −

* Activities
* Association
* Conditions
* Constraints

Once the above-mentioned parameters are identified, we need to make a mental layout of the entire flow. This mental layout is then transformed into an activity diagram.

Following is an example of an activity diagram for order management system. In the diagram, four activities are identified which are associated with conditions. One important point should be clearly understood that an activity diagram cannot be exactly matched with the code. The activity diagram is made to understand the flow of activities and is mainly used by the business users

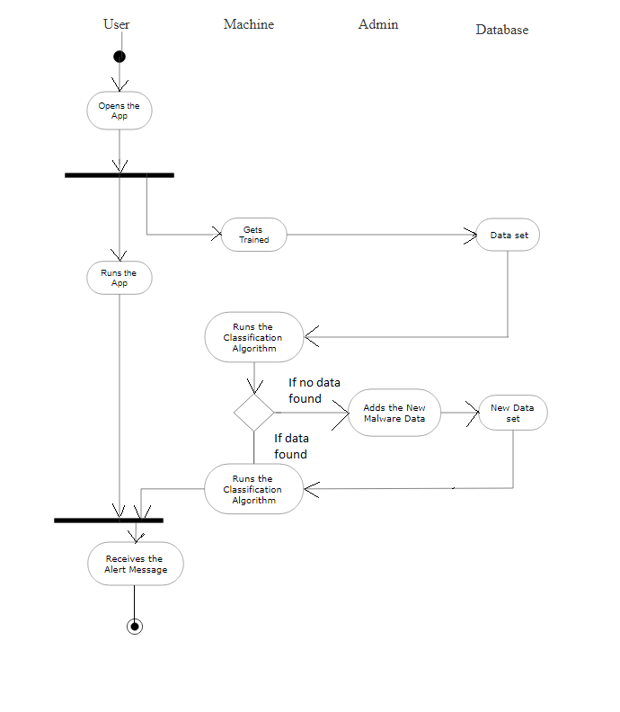


Figure 4.1.2 Activity Diagram

### 

### 

#### Sequence Diagrams:

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of a collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.

#### Sequence Diagrams captures:

* the interaction that takes place in a collaboration that either realizes a use case or an operation (instance diagrams or generic diagrams)
* high-level interactions between user of the system and the system, between the system and other systems, or between subsystems (sometimes known as system sequence diagrams)

#### Sequence Diagrams at a Glance:

Sequences Diagrams show elements as they interact over time and they organized according to object horizontally and time vertically.

#### Object Dimension:

* The horizontal axis shows the elements that are involved in the interaction
* Conventionally, the objects involved in the operation are listed from left to right according to when they take part in the message sequence. However, the elements on the horizontal axis may appear in any order

#### Sequence Diagram

The Sequence Diagram represents the sequence of events that takes place with respect to the timeline. They capture the interaction between objects in the context of a collaboration. Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.

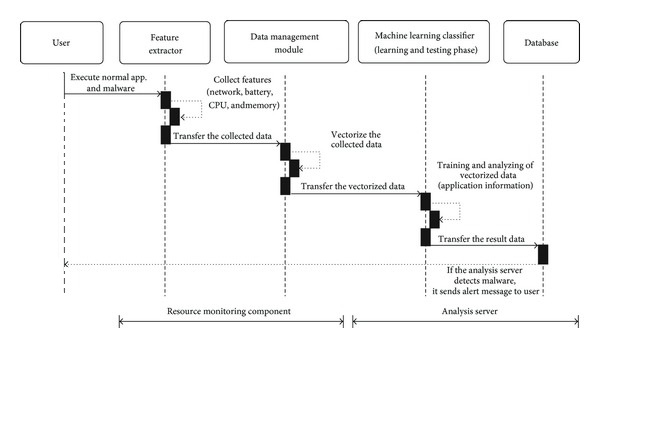


Figure 3.3.3.13 Sequence Diagram

In the above Sequence Diagram Registered User, Feature extractor, Data Management module, Machine Learning classifier, Database are life lines . Here, Execute normal app and malware, Transfer the collected data, Transfer the vectorized data, Transfer the result data are “Synchronus Messages”. Collect features,Vectorize the collected data, Training and analyzing of vectorized data are Self Maessages. lt, Receives new Data are the Returns new Data.

## PACKAGES USED IN THE PROGRAM:

#### PANDAS:

Pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with structured (tabular, multidimensional, potentially heterogeneous) and time series data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis / manipulation tool available in any language. It is already well on its way toward this goal.

Pandas is well suited for many different kinds of data:

* Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet
* Ordered and unordered (not necessarily fixed-frequency) time series data.
* Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels
* Any other form of observational / statistical data sets. The data actually need not be labeled at all to be placed into a pandas data structure

The two primary data structures of pandas, Series (1-dimensional) and DataFrame (2-dimensional), handle the vast majority of typical use cases in finance, statistics, social science, and many areas of engineering. For R users, DataFrame provides everything that R’s provides and much more. pandas is built on top of [NumPy](http://www.numpy.org/) and is intended to integrate well within a scientific computing environment with many other 3rd party libraries.

Here are just a few of the things that pandas does well:

* Easy handling of **missing data** (represented as NaN) in floating point as well as non-floating point data
* Size mutability: columns can be **inserted and deleted** from DataFrame and higher dimensional objects
* Automatic and explicit **data alignment**: objects can be explicitly aligned to a set of labels, or the user can simply ignore the labels and let Series, DataFrame, etc. automatically align the data for you in computations.
* Powerful, flexible **group by** functionality to perform split-apply-combine operations on data sets, for both aggregating and transforming data
* Make it **easy to convert** ragged, differently-indexed data in other Python and NumPy data structures into DataFrame objects.
* Intelligent label-based **slicing**, **fancy indexing**, and **subsetting** of large data sets
* Intuitive **merging** and **joining** data sets
* Flexible **reshaping** and pivoting of data sets
* **Hierarchical** labeling of axes (possible to have multiple labels per tick)
* Robust IO tools for loading data from **flat files** (CSV and delimited), Excel files, databases, and saving / loading data from the ultrafast **HDF5 format**
* **Time series**-specific functionality: date range generation and frequency conversion, moving window statistics, moving window linear regressions, date shifting and lagging, etc.

#### Sklearn:

[Scikit-learn](http://scikit-learn.org/stable/) is a library in Python that provides many unsupervised and supervised learning algorithms. It's built upon some of the technology you might already be familiar with, like NumPy, pandas, and Matplotlib!

The functionality that scikit-learn provides include:

* **Regression**, including Linear and Logistic Regression
* **Classification**, including K-Nearest Neighbors
* **Clustering**, including K-Means and K-Means++
* **Model selection**
* **Preprocessing**, including Min-Max Normalization
* [Dimensionality reduction](http://scikit-learn.org/stable/modules/decomposition.html#decompositions)

#### Matplotlib:

[matplotlib.pyplot](https://matplotlib.org/api/pyplot_api.html#module-matplotlib.pyplot) is a collection of command style functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc. In [matplotlib.pyplot](https://matplotlib.org/api/pyplot_api.html" \l "module-matplotlib.pyplot" \o "matplotlib.pyplot) various states are preserved across function calls, so that it keeps track of things like the current figure and plotting area, and the plotting functions are directed to the current axesIf matplotlib were limited to working with lists, it would be fairly useless for numeric processing. Generally, you will use [numpy](http://www.numpy.org/) arrays. In fact, all sequences are converted to numpy arrays internally.

matplotlib is also a massive library, and getting a plot to look just right is often achieved through trial and error. Using one-liners to generate basic plots in matplotlib is fairly simple, but skillfully commanding the remaining 98% of the library can be daunting.

Overall, Matplotlib is a python library used to create 2D graphs and plots by using python scripts. It has a module named pyplot which makes things easy for plotting by providing feature to control line styles, font properties, formatting axes etc. It supports a very wide variety of graphs and plots namely - histogram, bar charts, power spectra, error charts etc. It is used along with NumPy to provide an environment that is an effective open source alternative for MatLab. It can also be used with graphics toolkits like PyQt and wxPython.

Conventionally, the package is imported into the Python script by adding the following statement –

from matplotlib import pyplot as plt

#### SVM:

Support vector machines (SVMs) are a set of supervised learning methods used for [classification](http://scikit-learn.org/stable/modules/svm.html#svm-classification), [regression](http://scikit-learn.org/stable/modules/svm.html#svm-regression) and [outliers detection](http://scikit-learn.org/stable/modules/svm.html#svm-outlier-detection).

The advantages of support vector machines are:

* Effective in high dimensional spaces.
* Still effective in cases where number of dimensions is greater than the number of samples.
* Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
* Versatile: different [Kernel functions](http://scikit-learn.org/stable/modules/svm.html#svm-kernels) can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

The disadvantages of support vector machines include:

* If the number of features is much greater than the number of samples, avoid
* over-fitting in choosing [Kernel functions](http://scikit-learn.org/stable/modules/svm.html#svm-kernels) and regularization term is crucial.
* SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation (see [Scores and probabilities](http://scikit-learn.org/stable/modules/svm.html#scores-probabilities), below).

The support vector machines in scikit-learn support both dense (numpy.ndarray and convertible to that by numpy.asarray) and sparse (any scipy.sparse) sample vectors as input. However, to use an SVM to make predictions for sparse data, it must have been fit on such data. For optimal performance, use C-ordered numpy.ndarray (dense) orscipy.sparse.csr\_matrix (sparse) with dtype=float64.

Regression:

The method of Support Vector Classification can be extended to solve regression problems. This method is called Support Vector Regression.

The model produced by support vector classification (as described above) depends only on a subset of the training data, because the cost function for building the model does not care about training points that lie beyond the margin. Analogously, the model produced by Support Vector Regression depends only on a subset of the training data, because the cost function for building the model ignores any training data close to the model prediction.

There are three different implementations of Support Vector Regression: [SVR](http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html#sklearn.svm.SVR), [NuSVR](http://scikit-learn.org/stable/modules/generated/sklearn.svm.NuSVR.html" \l "sklearn.svm.NuSVR" \o "sklearn.svm.NuSVR) and [LinearSVR](http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVR.html" \l "sklearn.svm.LinearSVR" \o "sklearn.svm.LinearSVR). [LinearSVR](http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVR.html" \l "sklearn.svm.LinearSVR" \o "sklearn.svm.LinearSVR) provides a faster implementation than [SVR](http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html#sklearn.svm.SVR) but only considers linear kernels, while [NuSVR](http://scikit-learn.org/stable/modules/generated/sklearn.svm.NuSVR.html" \l "sklearn.svm.NuSVR" \o "sklearn.svm.NuSVR) implements a slightly different formulation than [SVR](http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html#sklearn.svm.SVR) and [LinearSVR](http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVR.html" \l "sklearn.svm.LinearSVR" \o "sklearn.svm.LinearSVR)

#### Classification:

SVC and [Nu SVC](http://scikit-learn.org/stable/modules/generated/sklearn.svm.NuSVC.html#sklearn.svm.NuSVC) are similar methods, but accept slightly different sets of parameters and have different mathematical formulations (see section [Mathematical formulation](http://scikit-learn.org/stable/modules/svm.html#svm-mathematical-formulation)). On the other hand, [Linear SVC](http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html#sklearn.svm.LinearSVC) is another implementation of Support Vector Classification for the case of a linear kernel. Note that [Linear SVC](http://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html#sklearn.svm.LinearSVC) does not accept keyword kernel, as this is assumed to be linear. It also lacks some of the members of [SVC](http://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#sklearn.svm.SVC) and [Nu SVC](http://scikit-learn.org/stable/modules/generated/sklearn.svm.NuSVC.html#sklearn.svm.NuSVC), like support.

#### PYTHON UTILS:

Python Utils is a collection of small Python functions and classes which make common patterns shorter and easier. It is by no means a complete collection but it has served me quite a bit in the past and I will keep extending it.

Example:

To extract a number from nearly every string:

from python\_utils import converters

number = converters.to\_int('spam15eggs')

assert number == 15

number = converters.to\_int('spam')

assert number == 0

number = converters.to\_int('spam', default=1)

assert number == 1

number = converters.to\_float('spam1.234')

#### TRAIN\_TEST\_SPLIT :

**Syntax**: sklearn.model\_selection.train\_test\_split(\*arrays,\*\*options)[source]

Split arrays or matrices into random train and test subsets

Quick utility that wraps input validation and also  next(ShuffleSplit().split(X, y)) and application to input data into a single call for splitting (and optionally subsampling) data in a oneliner.

|  |
| --- |
| **Arrays** : sequence of indexables with same length / shape[0]  Allowed inputs are lists, numpy arrays, scipy-sparse matrices or pandas dataframes.  test\_size : float, int or None, optional (default=0.25)   * If float, should be between 0.0 and 1.0 and represent the proportion of the dataset to include in the test split. If int, represents the absolute number of test samples. If None, the value is set to the complement of the train size. By default, the value is set to 0.25. The default will change in version 0.21. It will remain 0.25 only if train\_size is unspecified, otherwise it will complement the specified train\_size. * train\_size : float, int, or None, (default=None) * If float, should be between 0.0 and 1.0 and represent the proportion of the dataset to include in the train split. If int, represents the absolute number of train samples. If None, the value is automatically set to the complement of the test size. * random\_state : int, RandomState instance or None, optional (default=None) * If int, random\_state is the seed used by the random number generator; If RandomState instance, random\_state is the random number generator; If None, the random number generator is the RandomState instance used by np.random. * shuffle : boolean, optional (default=True) * Whether or not to shuffle the data before splitting. If shuffle=False then stratify must be None.   + - stratify : array-like or None (default=None) * If not None, data is split in a stratified fashion, using this as the class labels. |

## Sample Code

### Importing Preprocessor

import random

import pandas as pd

from bs4 import BeautifulSoup

import re

from nltk.corpus import stopwords

import matplotlib.pyplot as plt

from utils import getTags

def plotToWords(raw\_plot):

plot = BeautifulSoup(raw\_plot, "lxml")

letters\_only = re.sub("[^a-zA-Z]", "", plot.get\_text())

lower\_case = letters\_only.lower()

words = lower\_case.split()

stops = set(stopwords.words("english")

meaningful\_words = [w for w in words if not w in stops]

return ("".join(meaningful\_words))

def preprocess(filename):

train = pd.read\_csv(filename)

counts = train.Genre1.value\_counts()

counts.plot(kind='bar')

plt.show()

print (counts)

num\_reviews = train["Plot"].size

clean\_train\_reviews = []

for i in range(0, num\_reviews):

if ((i + 1) % 100 == 0):

print ("Review %d of %d\n" % (i + 1, num\_reviews))

clean\_train\_reviews.append(plotToWords(train["Plot"][i]))

tagVector = getTags('Comedy', train)

data = {'plot': clean\_train\_reviews, 'tags': tagVector}

df = pd.DataFrame(data)

return df

### CODE FOR SVM(SUPPORT VECTOR MACHINE):

from sklearn import svm

from sklearn.cross\_validation import train\_test\_split

from sklearn.feature\_extraction.text import CountVectorizer

from utils import predict

from preprocess import preprocess

data\_features = preprocess("C:\\AI container\\Python\\Project\\data\\trainingSet.csv")

train\_data, test\_data = train\_test\_split(data\_features, test\_size=0.1, random\_state=42)

print (len(test\_data))

vectorizer = CountVectorizer(analyzer="word", tokenizer=None, preprocessor=None, stop\_words=None, max\_features=3000)

train\_data\_features = vectorizer.fit\_transform(train\_data['plot'])

train\_data\_features = train\_data\_features.toarray()

lin\_clf = svm.LinearSVC()

lin\_clf.fit(train\_data\_features, train\_data['tags'])

predict(vectorizer, lin\_clf, test\_data)

### Code for Logistic Regression

from sklearn.cross\_validation import train\_test\_split

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn import linear\_model

from utils import predict

import matplotlib.pyplot as plt

from preprocess import preprocess

data\_features=preprocess("C:\\AIcontainer\\Python\\Project\\data\\trainingSet.csv")

train\_data, test\_data = train\_test\_split(data\_features, test\_size=0.1, random\_state=42)

counts = test\_data.tags.value\_counts()

counts.plot(kind='bar')

plt.show()

vectorizer = CountVectorizer(analyzer="word", tokenizer=None, preprocessor=None, stop\_words=None, max\_features=3000)

train\_data\_features = vectorizer.fit\_transform(train\_data['plot'])

train\_data\_features = train\_data\_features.toarray()

logreg = linear\_model.LogisticRegression(n\_jobs=1,

C=1e5)logreg = logreg.fit(train\_data\_features, train\_data['tags'])

predict(vectorizer, logreg, test\_data)

### Code Logistic Regression of Term Frequency and Inverse Document Frequency

from sklearn.cross\_validation import train\_test\_split

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn import linear\_model

from utils import predict

from preprocess import preprocess

data\_features = preprocess("C:\\AI container\\Python\\Project\\data\\trainingSet.csv")

train\_data, test\_data = train\_test\_split(data\_features, test\_size=0.1, random\_state=42)

tf\_vect=TfidfVectorizer(min\_df=2,tokenizer=None,preprocessor=None,stop\_words=None)

train\_data\_features = tf\_vect.fit\_transform(train\_data['plot'])

train\_data\_features = train\_data\_features.toarray()

logreg = linear\_model.LogisticRegression(n\_jobs=1, C=1e5)

logreg = logreg.fit(train\_data\_features, train\_data['tags'])

predict(tf\_vect, logreg, test\_data)

# RESULTS AND DISCUSSION

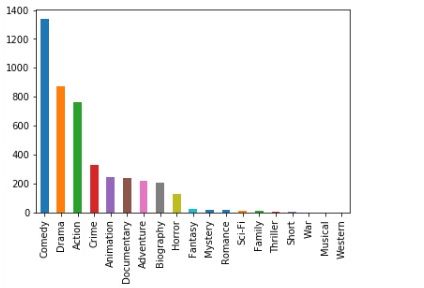


Figure 4.1 Screen-1

Figure 5.1 Showing the bar graph types and number of Genres respectively present in the dataset. We can see that the dataset has Comedy more than others Genres. The Dataset considered was IMDb. The Genres shown are from the Movies ,TV series, Web Series in the dataset. We can see that the "Western", "Musical", "War" Genre has the number.

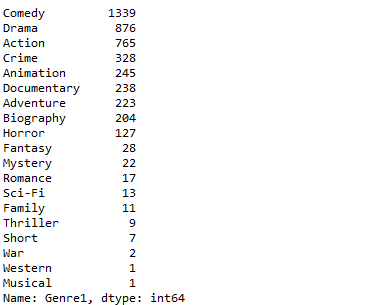


Figure 4.2 Screen-2

Figure 4.2 shows the number and types of Genres respectively in number. Comedy has the highest 1339 and the least War, Western and Musical has 2,1,1 in number respectively.

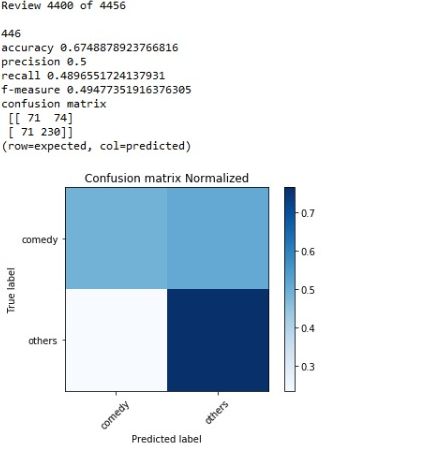


Figure 4.3 Screen-3

Figure 4.3 Screen shows a confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known. By the above Confusion matrix, the predicted value is checked against the true values can be known. We can see the accuracy of the Algorithm which is 67.48%(approx) and the Precision 0.5.

# CONCLUSIONS AND FUTURE SCOPE

## Conclusion

We can predict whether a movie belongs to a particular Genre or not using various data mining and Machine Learning processes. We have collected the data available on internet and compile a data set which will be primarily based on IMDB data. Now we have built a program which uses [NLTK](http://www.nltk.org/) and [Scikit-learn](http://scikit-learn.org/stable/) libraries. Our goal is to build a tool which is a classification model that predicts whether a movie belongs to a particular Genre or not which gives good accurate results.

## Future Scope

The major research scope areas by classification and Regression are:

* Image Recognition with more accuracy;
* Earthquake prediction;
* Medical Diagnosis;
* Statistical Arbitrage;
* Rise of Robots;
* Energy market price forecasting

# 

# 

# REFERENCES

[1] Domain information from the website [https://www.geeksforgeeks.org/regression-classification-supervised-machine-learning](https://www.geeksforgeeks.org/regression-classification-supervised-machine-learning/)

[2] NLTK and NLP description from [www.nltk.org](http://www.nltk.org)

# [3] Movie genre preference prediction using machine learning for customer-based information

[https://ieeexplore.ieee.org/document/8301647/references#references](%20https:/ieeexplore.ieee.org/document/8301647/references%23references)

[4] Predicting movies genres based on plot summaries <https://arxiv.org/abs/1801.04813>

[5] What is Machine Learning: <https://en.wikipedia.org/wiki/Machine_learning>

[6] For genre preference using machine learning for customer based information: [https://waset.org/publications/10008614/movie-genre-preference-prediction-using-machine-learning-for-customer-based-information](https://waset.org/publications/10008614/movie-genre-preference-prediction-using-machine-learning-for-customer-based-information-)

[7] Types of classification algorithms :[https://www.analyticsindiamag.com/7-types-classification-algorithms](https://www.analyticsindiamag.com/7-types-classification-algorithms/)

[8] How logistic regression can be used for classification tasks, and explores how to evaluate the effectiveness of classification models.<https://developers.google.com/machine-learning/crash-course/classification/video-lecture>

[9] Machine Learning in Python: <http://scikit-learn.org/stable/>

[10] Python for Machine Learning [https://pythonprogramming.net/machine-learning-tutorial-python-introduction](https://pythonprogramming.net/machine-learning-tutorial-python-introduction/)

**Bibliography:**

John, G.: ‘Netflix Case Study: David Becomes Goliath’, in Editor: ‘Book Netflix Case Study: David Becomes Goliath’ (2008, September 13, edn.)