**CHAPTER - 1**

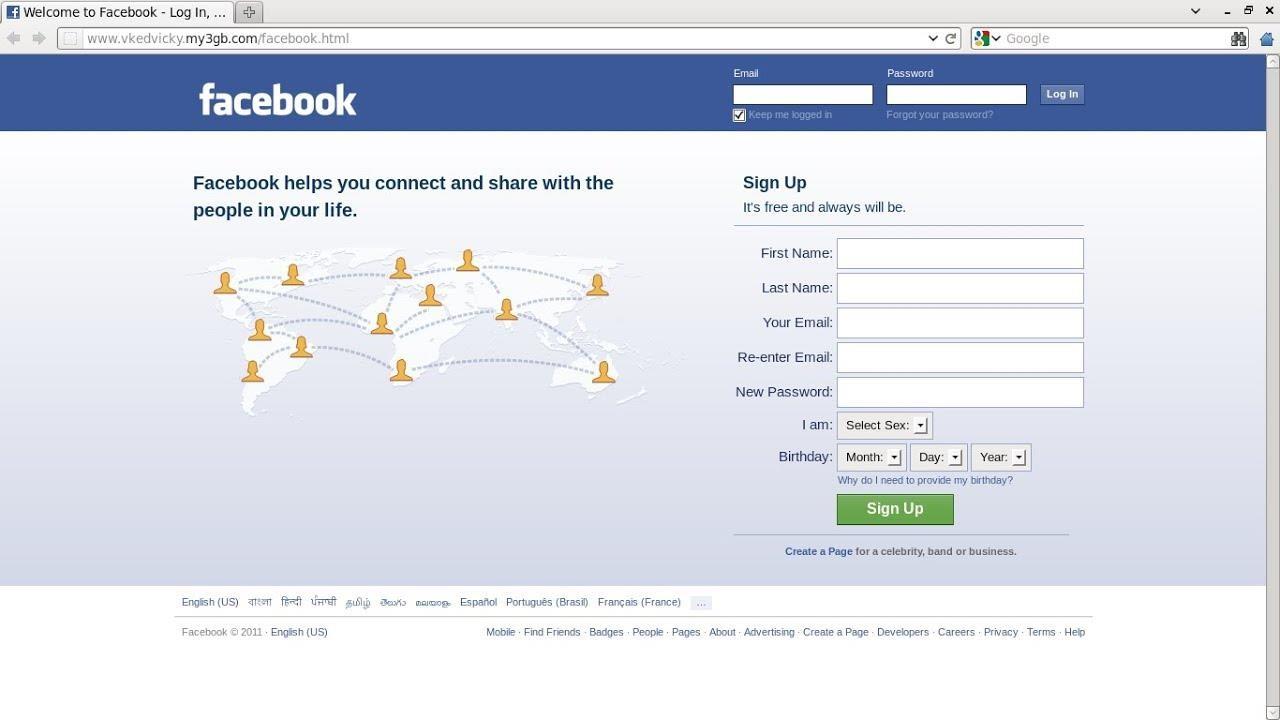
**INTRODUCTION**

1.1 About Project:

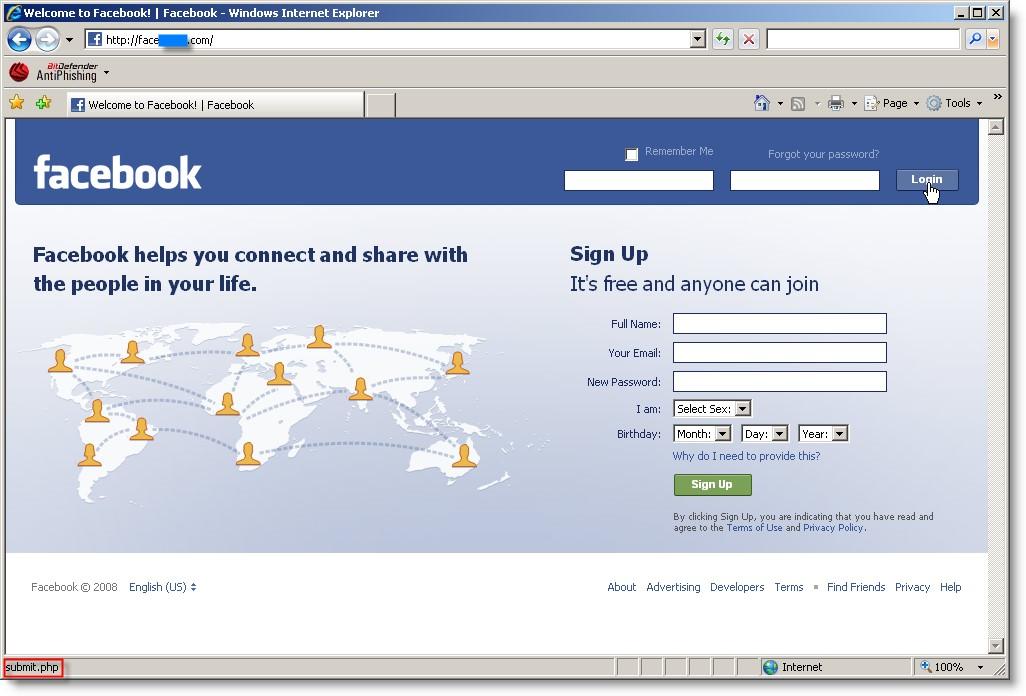
Phishing is a criminal mechanism employing both social engineering and technical tricks to steal consumers’ personal identity data and financial account credentials. Social engineering schemes use spoofed emails, purporting to be from legitimate businesses and agencies, designed to lead consumers to counterfeit websites that trick recipients into divulging financial data such as usernames and passwords. Technical subterfuge schemes install malicious software onto computers, to steal credentials directly, often using systems to intercept consumers’ online account usernames and passwords. Figure. 1 represents the webpage of the popular website www.facebook.com. Figure. 2 represents a webpage similar to that of facebook, but is the webpage of a site which spreads phishing activities. A user may misunderstand the second site as a genuine facebook site and provide his personal identity details. The Phisher can thus steal that information and he may use it for vicious purposes.

**A. The Technique of Phishing:**

The criminals, who want to obtain sensitive data, first create unauthorized replicas of a real website and email, usually from a financial institution or another company that deals with financial information. The email will be created using logos and slogans of a legitimate company.



**Figure1. Original face-book webpage**



**Figure 2. Phishing webpage**

Website creation is one of the reasons that the Internet has grown so rapidly as a communication medium, it also permits the abuse of trademarks, trade names, and other corporate identifiers upon which consumers have come to rely as mechanisms for authentication. Phisher then sends the "spoofed" emails to as many people as possible in an attempt to lure them into- the scheme. When these emails are opened or when a link in the mail is clicked, the consumers are redirected to a spoofed website, appearing to be from the legitimate entity.

**B. Statistics of Phishing attacks:**

Phishing continues to be one of the rapidly growing classes of identity theft scams on the internet that is causing both short term and long-term economic damage. There have been nearly 33,000 phishing attacks globally each month in the year 2020, totaling a loss of $80,183 million.

An example of phishingt[he Daily Swig reported](https://portswigger.net/daily-swig/data-breach-at-healthcare-provider-elara-caring-exposes-100-000-patients-information) a phishing attack that occurred in December 2020 at US healthcare provider Elara Caring that came after an unauthorized computer intrusion targeting two employees. The attacker gained access to the employees’ email accounts, resulting in the exposure of the personal details of over 100,000 elderly patients, including names, birth dates, financial and bank information, Social Security numbers, driver’s license numbers and insurance information.

##### 1.2 EXISTING SYSTEM DRAWBACKS:

* The general method to detect phishing websites by updating blacklisted URLs, Internet Protocol (IP) to the antivirus database which is also known as the “blacklist" method.
* To evade blacklists attackers use creative techniques to fool users by modifying the URL to appear legitimate via obfuscation and many other simple techniques.

**Disadvantages:**

* It cannot detect a Zero hour phishing attack.
* This technique fails when blacklisted URLs with minor changes are encountered.
* Less accuracy.
* No comprehensive blacklist can ensure a perfect up-to-date database.

##### 

##### 1.3 PROPOSED SYSTEM FEATURES:

* To overcome the drawbacks of blacklist and heuristics-based methods, many security researchers now focused on machine learning techniques.
* Machine learning technology consists of many algorithms which requires past data to make a decision or prediction on future data.
* Using this technique, the algorithm will analyze various blacklisted and legitimate URLs and their features to accurately detect the phishing websites including zero- hour phishing websites**.**

**Advantages:**

* Can detect zero hour phishing attacks.
* More customer retention.
* High accuracy.
* Cost is considered.
* Large Data can be used in machine learning.

**CHAPTER - 2**

**ANALYSIS**

**2.1 Hardware and Software Requirements**

**2.1.1 Hardware Requirements:**

Processor : Intel Core (i3/i5/i7)

RAM : 4.00GB

Hard Disk : 500GB

#### 2.1.2 Software Requirements:

Technology :Python 3.8.32

IDE :Jupyternotebook (Anaconda)

Operating system :Windows 7/8/10

Browser :Google chrome (51.2 and above)

**2.2 Functional Requirements:**

* The following are the functional requirements of our project.
* A training dataset has to be created on which training is performed.
* A testing dataset has to be created on which testing is performed

**2.3 Non-Functional Requirements:**

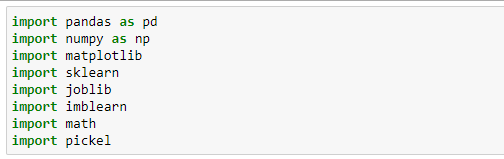
* **Maintainability**: Maintainability is used to make future maintenance easier, meet new requirements.
* **Robustness**: Robustness is the quality of being able to withstand stress, pressures or changes in procedure or circumstances.
* **Reliability:** Reliability is an ability of a person or system to perform and maintain its functions in circumstances.
* **Size:** The size of a particular application play a major role, if the size is less then efficiency will be high.
* **Speed:** If the speed is high then it is good. Since the number of lines in our code is less, hence the speed is high.

##### 2.4 Module Description:

1. **Data Gathering :**

Data gathering is the most important step in solving any supervised machine learning problem.

Firstly, we will define our libraries like NumPy, pandas, matplotlib, sklearn, in the code. NumPy is used for matrix multiplications, Pandas is used for data manipulation, Matplotlib for visualization of data and sklearn is for Machine learning Algorithms.



Next step is to import the URL dataset in the code by using the Pandas module. The URL dataset contains features like Bag-of-words, URL’s, Anchors, Meta tags and so on. After importing the data, we should check for the extreme values and missing values in all the features of the URL dataset.

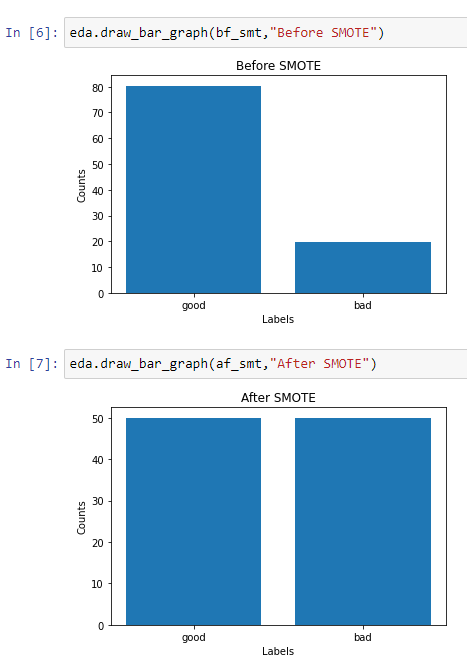
**2. Data Visualization:**

Data visualization is the presentation of data or information in a graph, chart or other visual format. It communicates relationships of data with images.

# 

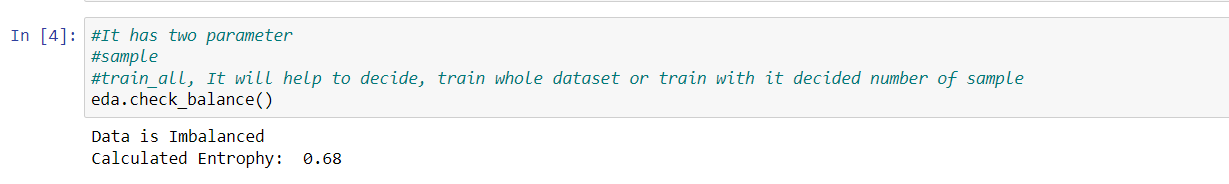
# EDA :

* Exploratory data analysis is an approach of analyzingdata sets to summarize their main characteristics, often using statistical graphics and other data visualization methods.
* A statistical model can be used or not, but primarily EDA is for seeing what the data can tell us beyond the formal modeling or hypothesis testing task.
* EDA is different from initial data analysis (IDA)[,[1]](https://en.wikipedia.org/wiki/Exploratory_data_analysis#cite_note-1)which focuses more narrowly on checking assumptions required for model fitting and hypothesis testing, and handling missing values and making transformations of variables as needed.
* EDA encompasses IDA.
* There are a number of tools and techniques that are useful for EDA, but EDA is characterized more by the attitude taken than by particular techniques.
* Typical graphical techniques used in EDA are:
* Box plot
* Histogram
* Multi-vari chart
* Run chart
* Pareto chart
* Scatter plot



**3. Data Preprocessing:**

* Data preprocessing is a process of preparing raw data and making it suitable for a machine learning model.
* It is a first and crucial step while creating a machine learning model.
* When creating a machine learning project, it is not always the case that we come across clean and formatted data.



# 

# 4. Feature Extraction:

The URL’s in the dataset includes multiple features like Bag-of-words, URL’s, Anchors, Meta tags and so on. By using all those features, the accuracy of the model to predict the given URL whether good or bad is low. To increase the accuracy of the model we should have to select the important or relevant features by using which we can get the accurate prediction of the URL to which item either good or bad it belongs to.

Feature extraction is a process of dimensionality reduction by which an initial set of raw data is reduced to more manageable groups for processing. The process of feature extraction is useful when you need to reduce the number of resources needed for processing without losing important or relevant information.

**FEATURE GROUPS**

|  |  |  |
| --- | --- | --- |
| **Features** | |  |
| JS-Enable-Disable | DNS PTR record | Path tokens |
| Document Frequency DF | WHOIS info | Last token of the path |
| Title tag <title>??</title> | Connection speed | Spamassassin plugin |
| 3-4-5 grams | TLD + domain | TLD |
| TF-IDF weighting | DNS A record | DNS TTL |
| Blacklists | Geographic | DNS MX record |
| WHOIS dates | Hostname | Bag-of-words |
| IP address misc | Words+URLs | URLs |
| Lexical misc | Meta+link | Anchors |
| 4 grams | URLs+anchors | Meta tags |
| URLs+anchors+meta | |  |

**BAG OF WORDS:**

Bag of Words (BOW) is a method to extract features from text documents. These features can be used for training machine learning algorithms. It creates a vocabulary of all the unique words occurring in all the documents in the training set.

**TF-IDF:**

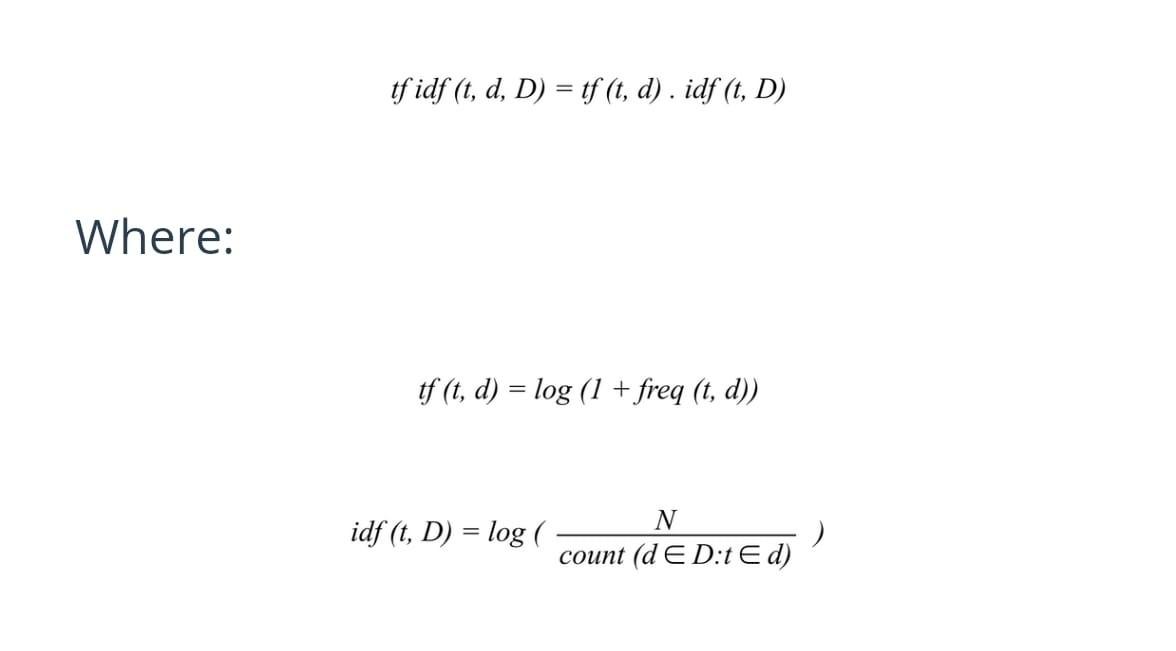
TF-IDF is a statistical measure that evaluates how relevant a word is to a document in a collection of documents. This is done by multiplying two metrics: how many times a word appears in a document, and the inverse document frequency of the word across a set of documents.

TF-IDF (term frequency-inverse document frequency) was invented for document search and information retrieval. It works by increasing proportionally to the number of times a word appears in a document, but is offset by the number of documents that contain the word.

**How is TF-IDF calculated?**

* TF-IDF for a word in a document is calculated by multiplying two different metrics:
* The term frequency of a word in a document.
* There are several ways of calculating this frequency, with the simplest being a raw count of instances a word appears in a document.
* Then, there are ways to adjust the frequency, by length of a document, or by the raw frequency of the most frequent word in a document.
* So, if the word is very common and appears in many documents, this number will approach 0. Otherwise, it will approach 1.
* Multiplying these two numbers results in the TF-IDF score of a word in a document.
* The higher the score, the more relevant that word is in that particular document.

To put it in more formal mathematical terms, the TF-IDF score for the word t in the document d from the document set D is calculated as follows:



**Why is TF-IDF used in Machine Learning?**

Machine learning with natural language is faced with one major hurdle.

Once words are transformed into numbers, in a way that machine learning algorithms can understand, the TF-IDF score can be fed to algorithms such as Naive Bayes and Support Vector Machines, greatly improving the results of more basic methods like word counts.

Benefits of feature extraction are:

1. Reduces Overfitting
2. Improves Accuracy
3. Reduces Training Time

# 5. Applying GAUSSIAN NAIVE BAYESIAN Algorithms:

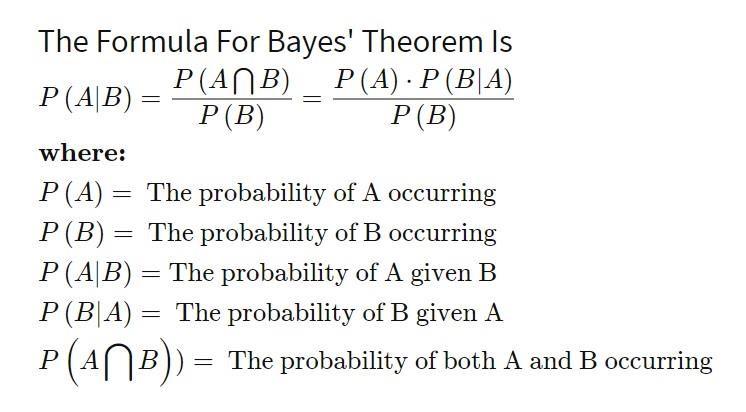
[**Gaussian Naive Bayes**](https://iq.opengenus.org/gaussian-naive-bayes/) is a variant of[Naive Bayes](https://iq.opengenus.org/text-classification-naive-bayes/)that follows Gaussian normal distribution and supports continuous data. We have explored the idea behind Gaussian Naive Bayes along with an example.

Before going into it, we shall go through a brief overview of Naive Bayes.

**Naive Bayes** are a group of supervised machine learning classification algorithms based on the **Bayes theorem**. It is a simple classification technique, but has high functionality. They find use when the dimensionality of the inputs is high. Complex classification problems can also be implemented by using Naive Bayes Classifier.

## Bayes Theorem:

Bayes Theorem can be used to calculate conditional probability. Being a powerful tool in the study of probability, it is also applied in Machine Learning.



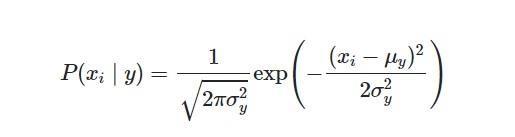
## 

## Naive Bayes Classifier:

Naive Bayes Classifiers are based on the Bayes Theorem. One assumption taken is the strong independence assumptions between the features. These classifiers assume that the value of a particular feature is independent of the value of any other feature. In a supervised learning situation, Naive Bayes Classifiers are trained very efficiently.

## Gaussian Naive Bayes:

When working with continuous data, an assumption often taken is that the continuous values associated with each class are distributed according to a normal (or Gaussian) distribution. The likelihood of the features is assumed to be-

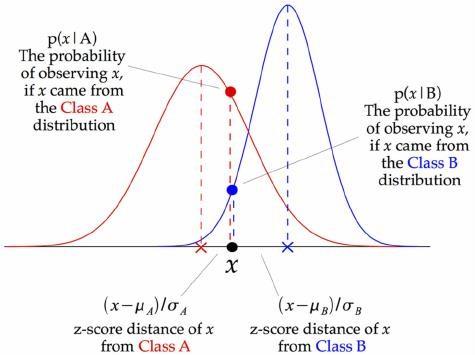


Sometimes assume variance

* is independent of Y (i.e., σi),
* or independent of Xi (i.e., σk)
* or both (i.e., σ)

Gaussian Naive Bayes supports continuous valued features and models each as conforming to a Gaussian (normal) distribution.

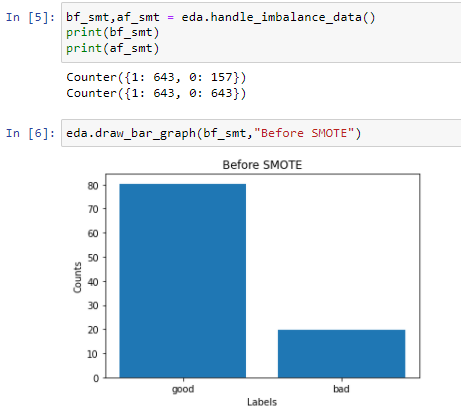
The data is described by a Gaussian distribution with no co-variance (independent dimensions) between dimensions. This model can be fit by simply finding the mean and standard deviation of the points within each label, which is all that is needed to define such a distribution.

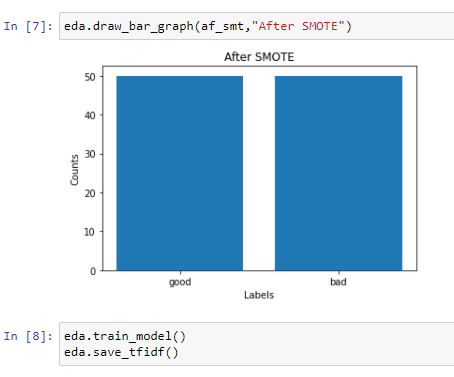


# 6. Accuracy and Performance:

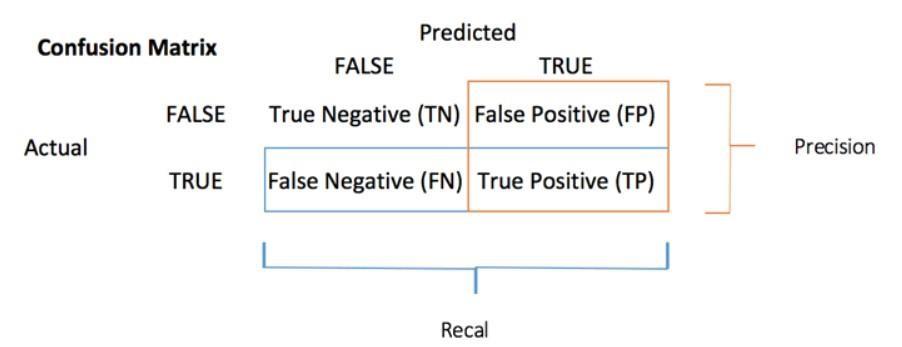
**SMOTE:**

The Synthetic Minority Oversampling (SMOTE) technique is used to increase the number of less presented cases in a data set used for machine learning. This is a better way to increase the number of cases than to simply duplicate existing cases.





**A confusion matrix** is a performance measurement method for Machine learning classification. It helps you to know the performance of the classification model on a set of test data so that the true values and false are known



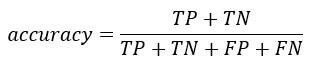
So, there are net 4 types of outcomes possible.

* **TP: True Positive**: Predicted values correctly predicted as actual positive
* **FP: False Positive**: Predicted values incorrectly predicted an actual positive. i.e.,

Negative values predicted as positive

* **FN: False Negative**: Positive values predicted as negative
* **TN: True Negative**: Predicted values correctly predicted as an actual negative We

can compute the accuracy test from the confusion matrix:



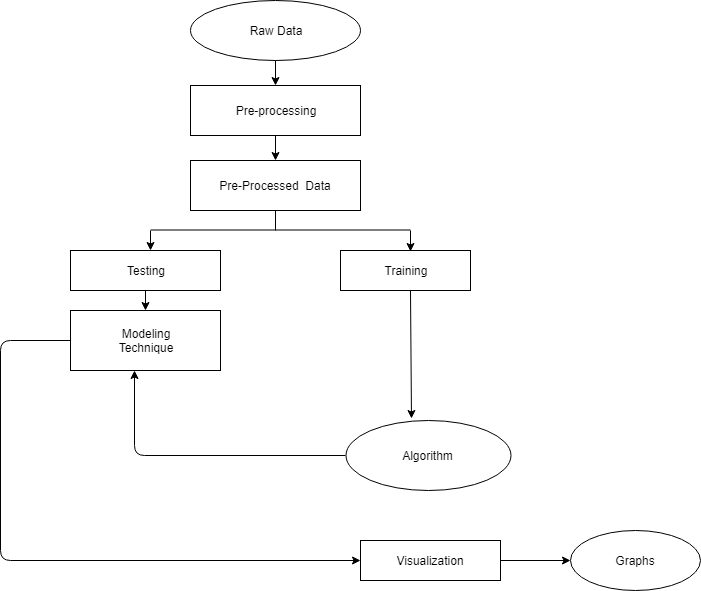
**CHAPTER -3**

**DESIGN**

System design is the process of defining the architecture, modules, interfaces and datafor a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product management.

##### BLOCK DIAGRAM:

The block diagram is typically used for a higher level, less detailed description aimed more at understanding the overall concepts and less at understanding the details of implementation.



##### DATA FLOW DIAGRAMS:

Data flow diagram (DFD) is a graphical representation of “flow” of data through an information system, modelling its process concepts. Often, they are a preliminary step used to create an overview of the system which can later be elaborated. DFD’s can also be used for the visualization of data processing (structured design).

A DFD shows what kinds of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It doesn’t show information about timing of processes, or information about whether processes will operate in sequence or parallel. A DFD is also called a “bubble chart”.

##### DFD Symbols:

In the DFD, there are four symbols:

* quire defines a source or destination of system data.
* An arrow indicates dataflow. It is the pipeline through which the information flows.
* A circle or a bubble transformsdata flow into outgoing dataflow.
* An open rectangle is a store, data at reset or at temporary repository of data.

**Dataflow:** Data moves in a specific direction from an origin to a destination.

**Process:** People, procedures or devices that use or produce (Transform) data. The physical component is not identified.

**Sources**: External sources or destination of data, which may be programs, organizations or other entities.

**Data store:** Here data is stored or referenced by a process in the systems.

In our project, we had built the data flow diagrams at the very beginning of business process modelling in order to model the functions that our project has to carry out and the interaction between those functions together with focusing on data exchanges between processes.

##### UNIFIED MODELLING LANGUAGE DIAGRAMS:

The Unified Modelling Language (UML) is a Standard language for specifying, visualizing, constructing and documenting the software system and its components. The UML focuses on the conceptual and physical representation of the system. It captures the decisions and understandings about systems that must be constructed. A UML system is represented using five different views that describe the system from a distinctly different perspective. Each view is defined by a set of diagrams, which is as follows.

* **User Model View** 
  + - * 1. This view represents the system from the user’s perspective.
        2. The analysis representation describes a usage scenario from the end user's perspective.
* **Structural Model View** 
  + - * 1. In this model the data and functionality are arrived from inside the system.
        2. This model view models the static structures.
* **Behavioral Model View**

It represents the dynamic of behavior as parts of the system, depicting the interactions of collection between various structural elements described in the user model and structural model view.

* **Implementation model View**

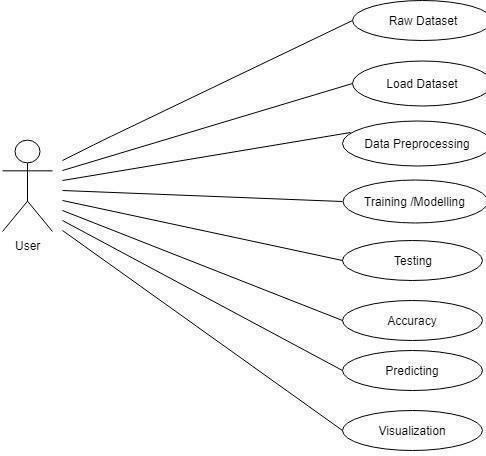
In this the structural and behavioral parts of the system are represented as they are to be built.

* **Environmental Model View**

In this the structural and behavioral aspect of the environment in which the system is to be implemented are represented.

##### 3.1 Use Case Diagram :

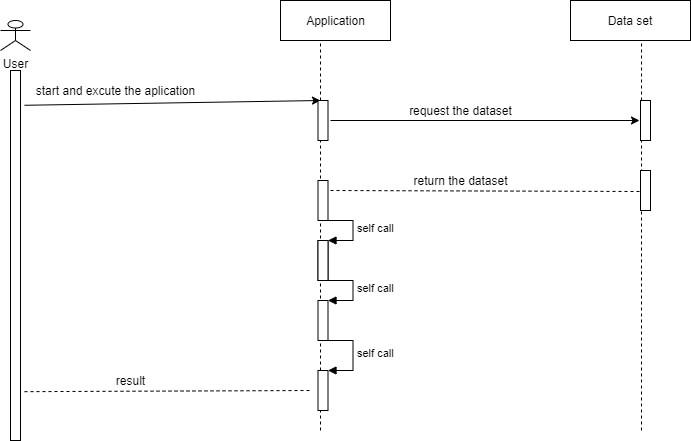
Use case diagrams are one of the five diagrams in the UML for modelling the dynamic aspects of the systems (activity diagrams, sequence diagram, state chart diagram, collaboration diagram are the four other kinds of diagrams in the UML for modelling the dynamic aspects of systems).Use case diagrams are central to modelling the behavior of the system, a sub-system, or a class. Each one shows a set of use cases and actors and relations.



##### Use case Diagram

**3.2 Sequence Diagram:**

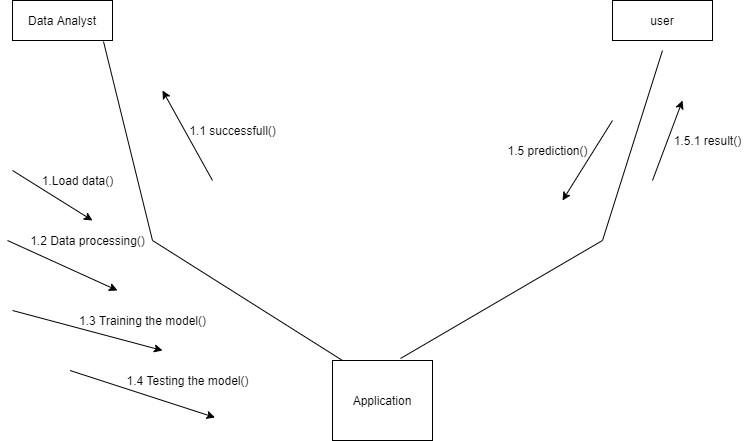
Sequence diagrams are a popular dynamic modelling solution. Dynamic modelling focuses on the interactions occurring within the system. Sequence diagrams specifically focus on the "lifelines" of an object and how they communicate with other objects to perform a function before the lifeline ends.



**SEQUENCE DIAGRAM**

**3.3 Collaboration Diagram:**

Collaboration diagram is an interaction diagram that emphasizes the structural organization of the objects that send and receive messages. Collaboration diagram and sequence diagram are isomorphic*.*



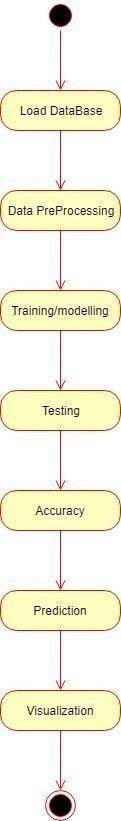
## 

## 

## 

## 3.3.4 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.



## CHAPTER - 4

## IMPLEMENTATION

Implementation is the stage of the project when the theoretical design is turned into a new and in giving the user confidence that the new system will work and be effective.

The implementation stage involves careful planning, investigation of the existing system and it’s constraints on implementation, designing of methods to achieve changeover and evaluating the working system. Thus it can be considered to be the most critical stage in achieving a successful non of changeover methods.

The project is implemented by accessing simultaneously from more than one system and more than one window in one system. The application is implemented in the Internet Information Services 5.0 web server under Windows XP and accessed from various clients.

##### Technologies Used What is Python?

**Python** is an interpreter, high-level programming language for general-purpose programming by “Guido van Rossum” and first released in 1991, Python has a design philosophy that emphasizes code readability, and a syntax that allows programmers to express concepts in fewer lines of code, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object- oriented, imperative, functional, procedural, and has a large and comprehensive standard library.

Python interpreters are available for many operating systems. Python, the reference implementation of Python, is open-source software and has a community-based development model, as do nearly all of its variant implementations. Python is managed by the non-profit Python Software Foundation.

Python is a general purpose, dynamic, high level and interpreted programming language. It supports object-oriented programming approaches to develop applications. It is simple and easy to learn and provides lots of high-level data structures.

* Windows XP
* Python Programming
* Open-source libraries: Pandas, NumPy, SciPy, matplotlib, OpenCV

##### Python Versions:

Python 2.0 was released on 16 October 2000 and had many major new features, including a cycle-detecting, garbage collector, and support for Unicode. With this release, the development process became more transparent and community-backed.

Python 3.0 (initially called Python 3000 or py3k) was released on 3 December 2008 after a long testing period. It is a major revision of the language that is not completely backward-compatible with previous versions. However, many of its major features have been back ported to the Python 2.6.x and 2.7.x version series, and releases of Python 3 include the 2to3 utility, which automates the translation of Python 2 code to Python 3.

Python 2.7's end-of-life date (a.k.a. EOL, sunset date) was initially set at 2015, then postponed to 2020 out of concern that a large body of existing code could not easily be forward-ported to Python 3. In January 2017, Google announced work on a Python 2.7 to Go Trans compiler to improve performance under concurrent workloads.

Python 3.6 had changes regarding UTF-8 (in Windows, PEP 528 and PEP 529) and P[ython 3.7.0b1 (PEP 540)](https://www.python.org/dev/peps/pep-0540/) added a new "UTF-8 Mode" (and overrides [POSIX locale).](https://en.wikipedia.org/wiki/POSIX_locale)

##### WHY PYTHON?

* Python is a scripting language like PHP, Perl, and Ruby.
* No licensing, distribution, or development fees
* It is a Desktop application.
* Linux, windows
* Excellent documentation
* Thriving developer community
* For us job opportunity

##### Libraries Of python:

Python's large standard library, commonly cited as one of its greatest strengths, provides tools suited too many tasks. For Internet-facing applications, many standard formats and protocols such as MIME and HTTP are supported. It includes modules for creating graphical user interfaces, connecting to relational databases, generating pseudorandom numbers, arithmetic with arbitrary precision decimals, manipulating regular expressions, and unit testing.

Some parts of the standard library are covered by specifications (for example, the Web Server Gateway Interface (WSGI) implementation wsgiref follows PEP 33), but most modules are not.

They are specified by their code, internal documentation, and test suites (if supplied). However, because most of the standard library is cross-platform Python code, only a few modules need altering or rewriting for variant implementations.

As of March 2018, the Python Package Index (PyPI), the official repository for third party Python software, contains over 130,000 packages with a wide range of functionality, including:

* Graphical user interfaces
* Web frameworks
* Multimedia
* Databases
* Networking
* Test frameworks
* Automation
* Web scraping
* Documentation
* System administration

##### What is Machine Learning?

Machine Learning is an application of artificial intelligence (AI) that provides system the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it to learn for themselves.

##### Basics of python machine learning:

* You'll know how to use Python and its libraries to explore your data with the help of matplotlib and Principal Component Analysis (PCA).
* And you'll pre-processes your data with normalization and you'll split your data into t raining and test sets.
* Next, you'll work with the well-known K-Means algorithm to construct an unsupervised model, fit this model to your data, predict values, and validate the model that you have built.
* As an extra, you'll also see how you can also use Support Vector Machines (SVM) to construct another model to classify your data.

##### Why Machine Learning?

* It was born from pattern recognition and theory that computers can learn without being programmed to specific tasks.
* It is a method of Data analysis that automates analytical model building.

Machine learning tasks are typically classified into two broad categories, depending on whether there is a learning "signal" or "feedback" available to a learning system. They are

[**Supervised learning:**](https://en.wikipedia.org/wiki/Supervised_learning)The computer is presented with example inputs and their desired outputs, given by a "teacher", and the goal is to learn a general rule that [maps](https://en.wikipedia.org/wiki/Map_(mathematics)) inputs to outputs. As special cases, the input signal can be only partially available, or restricted to special feedback:

[**Semi-supervised learning:**](https://en.wikipedia.org/wiki/Semi-supervised_learning)the computer is given only an incomplete training signal: a training set with some (often many) of the target outputs missing.

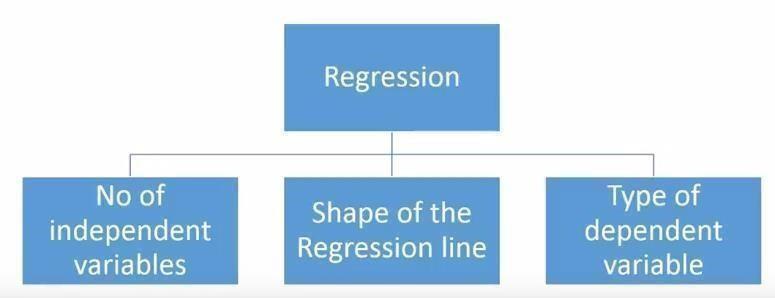
[**Active learning:**](https://en.wikipedia.org/wiki/Active_learning_(machine_learning)) the computer can only obtain training labels for a limited set of instances (based on a budget), and also has to optimize its choice of objects to acquire labels for. When used interactively, these can be presented to the user for labelling.

[**Reinforcement learning:**](https://en.wikipedia.org/wiki/Reinforcement_learning) training data (in form of rewards and punishments) is given only as feedback to the program's actions in a dynamic environment, such as [driving a vehicle](https://en.wikipedia.org/wiki/Autonomous_car) or playing a game against an opponent.

[**Unsupervised learning:**](https://en.wikipedia.org/wiki/Unsupervised_learning) No labels are given to the learning algorithm, leaving it on its own to find structure in its input. Unsupervised learning can be a goal in itself (discovering hidden patterns in data) or a means towards an end ([feature learning)](https://en.wikipedia.org/wiki/Feature_learning).

In [regression,](https://en.wikipedia.org/wiki/Regression_analysis) also a supervised problem, the outputs are continuous rather than discrete.

**Regression**: The analysis or measure of the association between one variable (the dependent variable) and one or more other variables (the independent variables), usually formulated in an equation in which the independent variables have parametric coefficients, which may enable future values of the dependent variable to be predicted.



**Regression Structure**

**Module:**

A module allows you to logically organize your Python code. Grouping related code into a module makes the code easier to understand and use. A module is a Python object with arbitrarily named attributes that you can bind and reference.

##### Pandas:

**Pandas** is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labelled” 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 labelled at all to be placed into a panda’s data structure

The two primary data structures of pandas, [Series](https://pandas.pydata.org/pandas-docs/stable/generated/pandas.Series.html#pandas.Series)(1-dimensional) and Data Frame (2 dimensional), handle the vast majority of typical use cases in finance, statistics, social science, and many areas of engineering. For R users, [Data Frame](https://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.html#pandas.DataFrame) provides everything that R’s data frame provides and much more. Pandas is built on top of NumPy and is intended to integrate well within a scientific computing environment with many other 3rd party libraries. 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 Data Frame 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*, *Data Frame*, 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 Data Frame objects
* Intelligent label-based slicing**,** fancy indexing, and sub setting of large data sets
* Intuitive merging and joining data sets
* Flexible reshaping and pivoting of data sets
* Hierarchical labelling 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.

Many of these principles are here to address the shortcomings frequently experienced using other languages / scientific research environments. For data scientists, working with data is typically divided into multiple stages: mugging and cleaning data, analyzing / modelling it, then organizing the results of the analysis into a form suitable for plotting or tabular display. pandas is the ideal tool for all of these tasks.

* Pandas is fast**.** Many of the low-level algorithmic bits have been extensively improved in Python code. However, as with anything else generalization usually sacrifices performance. So, if you focus on one feature for your application you may be able to create a faster specialized tool.
* Pandas are a dependency of stats models, making it an important part of the statistical computing ecosystem in Python.
* Pandas have been used extensively in production in financial applications.

##### NumPy: -

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. This tutorial explains the basics of NumPy such as its architecture and environment. It also discusses the various array functions, types of indexing, etc. An introduction to Matplotlib is also provided.

All this is explained with the help of examples for better understanding.

NumPy is a Python package. It stands for 'Numerical Python'. It is a library consisting of multidimensional array objects and a collection of routines for processing of arrays. Numeric, the ancestor of NumPy, was developed by Jim Humulin. Another package Numara was also developed, having some additional functionalities. In 2005, Travis Oliphant created the NumPy package by incorporating the features of Numara into Numeric package. There are many contributors to this open-source project.

##### Operations using NumPy: -

Using NumPy, a developer can perform the following operations –

* + Mathematical and logical operations on arrays.
  + Fourier transforms and routines for shape manipulation.
  + Operations related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.

**NumPy** – A Replacement for MATLAB

NumPy is often used along with packages like SciPy (Scientific Python) and Mat−plotid (plotting library). This combination is widely used as a replacement for MATLAB, a popular platform for technical computing. However, Python alternative to MATLAB is now seen as a more modern and complete programming language. It is open source, which is an added advantage of NumPy.

##### Scikit-learn: -

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

The scikit-learn project started as scikits. learn, a Google Summer of Code project by David Courmayeur. Its name stems from the notion that it is a “SciKit” (SciPy Toolkit), a separately-developed and distributed third-party extension to SciPy.

The original codebase was later rewritten by other developers. In 2010 Fabian Pedrosa, Gael Viroqua, AlexandreGramfort and Vincent Michel, all from INRIA took leadership of the project and made the first public release on February the 1st 2010. Of the various scikits, scikit-learn as well as scikit-image were described as “well-maintained and popular” in November 2012. Scikit-learn is largely written in Python, with some core algorithms written in Python to achieve performance. Support vector machines are implemented by a Python wrapper around LIBSVM; logistic regression and linear support vector machines by a similar wrapper around LIBLINEAR.

Some popular groups of models provided by scikit-learn include:

* **Ensemble methods:** for combining the predictions of multiple supervised models.
* **Feature extraction**: for defining attributes in image and text data.
* **Feature selection:** for identifying meaningful attributes from which to create supervised models.
* **Parameter Tuning:** for getting the most out of supervised models.
* **Manifold Learning:** For summarizing and depicting complex multi-dimensional data.
* **Supervised Models:** a vast array not limited to generalize linear models, discriminant analysis, naive bayes, lazy methods, neural networks, support vector machines and decision trees.

##### Matplotlib:-

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

# CHAPTER - 5

# CODING

**Phishing Websites**

**Importing Libraries**

**1.Gathering of Data:**

**In[]:**

# pandas - FILE Accessing and data frames.

**import pandas as pd**

#numpy - For numerical data (mathematical and logical operations).

**importnumpy as np**

# matplotlib-data visualization & graphical plotting.

# matplotlib.pyplot-pyplot is a function which creates figures,decorates a plot.

**importmatplotlib.pyplot as plt**

#joblib - set of tools to provide lightweight pipelining

#set of supervised machine learning classification algorithms.

**fromsklearn.naive\_bayes import GaussianNB**

# Reading Credit dataset into dataframe as a

In[]:

classPhishingModel():

def \_\_init\_\_(self,file\_name):

self.url\_data=pd.read\_csv(file\_name,error\_bad\_lines=False).sample(frac=1).reset\_index(drop=True)

self.bad\_data = self.url\_data[self.url\_data['label']=='bad']

self.good\_data = self.url\_data[self.url\_data['label']=='good']

self.preprocess\_data = []

self.labels=[]

self.tfidf = TfidfVectorizer(tokenizer=self.pre\_process\_url)

self.x\_train = None

self.y\_train = None

self.x\_test = None

self.y\_test = None

self.count=None

self.model = GaussianNB()

self.sm = SMOTE(random\_state=2)

**2. Data Visualizations**

defcreate\_graph(self,labels,counts,title):

plt.title(title)

plt.bar(labels,counts)

plt.xlabel("Labels")

plt.ylabel("Counts")

plt.plot()

defmodel\_bar\_graph(self,urls):

labels = np.unique(self.url\_data['label'].values)

counts = []

for x in labels:

counts.append(len(self.url\_data[self.url\_data['label']==x].values))

total = sum(counts)

fori in range(len(counts)):

counts[i] = round(((counts[i]/total)\*100),2)

self.create\_graph(labels,counts,"URL count")

defdraw\_bar\_graph(self,url,title):

labels = []

total=0

counts=[]

for x in url:

total+=url[x]

counts.append(url[x])

if(x==1):

labels.append("good")

else:

labels.append("bad")

fori in range(len(counts)):

counts[i] = round(((counts[i]/total)\*100),2)

self.create\_graph(labels,counts,title)

defcheck\_balance(self):

entropy = 0.0

labels = np.unique(self.url\_data['label'].values)

counts = []

for x in labels:

counts.append(len(self.url\_data[self.url\_data['label']==x].values))

entropy = 0.0

total = sum(counts)

for x in counts:

entropy += (-(x/total)\*np.log(x/total))

shannon\_en = entropy/np.log(len(counts))

shannon\_en = round(shannon\_en,3)

if(shannon\_en> 0.8):

print("Data is Balanced")

print("Calculated Entrophy: ",shannon\_en)

else:

print("Data is Imbalanced")

print("Calculated Entrophy: ",shannon\_en)

**3.Data Preprocessing:**

defpre\_process\_url(self,url):

data = str(url.encode('utf-8')).split('/')

total\_token = []

for x in data:

token = str(x).split('-')

token\_by\_dot=[]

for y in token:

temp\_token = str(y).split('.')

token\_by\_dot+=temp\_token

total\_token =total\_token+token+ token\_by\_dot

total\_token = list(set(total\_token))

if 'com' in total\_token:

total\_token.remove('com')

fori in range(len(total\_token)):

total\_token[i] = re.sub(r"[b']+",'',total\_token[i])

if '' in total\_token:

total\_token.remove('')

returntotal\_token

**4 .Feature Extraction:**

# Applyingtfidf

# Reducing features

defsave\_tfidf(self):

pickle.dump(self.tfidf, open("tfidf.pickle", "wb"))

defprocess\_data(self,sample=20,train\_all=0):

c=0

url = self.url\_data['url']

label = self.url\_data['label']

if(train\_all==1):

sample = int(0.05\*len(self.url\_data))

check\_data=[]

fori in range(sample):

check\_data.append(url[i])

self.preprocess\_data = self.tfidf.fit\_transform(check\_data)

for y in label:

if(y=='good'):

self.labels.append(1)

else:

self.labels.append(0)

c+=1

if(c==sample):

break

self.x\_train, self.x\_test, self.y\_train, self.y\_test = train\_test\_split(self.preprocess\_data,self.labels,test\_size=0.2,random\_state=42)

defhandle\_imbalance\_data(self):

before\_smote = Counter(self.y\_train)

self.x\_train,self.y\_train = self.sm.fit\_resample(self.x\_train,self.y\_train)

after\_smote = Counter(self.y\_train)

returnbefore\_smote,after\_smote

deftrain\_model(self):

self.model.fit(self.x\_train.toarray(),self.y\_train)

joblib.dump(self.model,"train\_model.pkl")

# 5. Accuracy and Performance

deftest\_accuracy(self):

self.model= joblib.load('train\_model.pkl')

predicted\_label = self.model.predict(self.x\_test.toarray())

acc = accuracy\_score(self.y\_test,predicted\_label)\*100

print(str(acc)+'%')

defprint\_confusion\_matrix(self):

self.model= joblib.load('train\_model.pkl')

predicted\_label = self.model.predict(self.x\_test.toarray())

data = confusion\_matrix(self.y\_test,predicted\_label)

df\_cm = pd.DataFrame(data, columns=np.unique(self.y\_test), index = np.unique(self.y\_test))

df\_cm.index.name = 'Actual'

df\_cm.columns.name = 'Predicted'

plt.figure(figsize = (10,7))

sns.set(font\_scale=1.4)#for label size

sns.heatmap(df\_cm, cmap="Blues", annot=True,annot\_kws={"size": 16})

defprint\_classification\_report(self):

self.model= joblib.load('train\_model.pkl')

predicted\_label = self.model.predict(self.x\_test.toarray())

print(classification\_report(self.y\_test,predicted\_label))

defpredict\_type(self,url):

self.model= joblib.load('train\_model.pkl')

self.tfidf = pickle.load(open("tfidf.pickle", "rb"))

prep = self.tf idf.transform([url])

flag = self.model.predict(prep.toarray())

if(flag[0]==1):

return "Url is Good"

else:

return "Url is Bad"

### CHAPTER - 6

### TESTING

It is the process of testing the functionality and it is the process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an undiscovered error. A successful test is one that uncovers an undiscovered error. Software testing is usually performed for one of two reasons:

* Defect Detection
* Reliability estimation

##### BLACK BOX TESTING:

The base of the black box testing strategy lies in the selection of appropriate data as per functionality and testing it against the functional specifications in order to check for normal and abnormal behavior of the system. Nowadays, it is becoming easier to route the testing work to a third party as the developer of the system knows too much of the internal logic and coding of the system, which makes it unfit to test applications by the developer. The following are different types of techniques involved in black box testing. They are:

* Decision Table Testing
* All pairs testing
* State transition tables testing
* Equivalence Partitioning

Software testing is used in association with Verification and Validation. Verification is the checking of or testing of items, including software, for conformance and consistency with an associated specification. Software testing is just one kind of verification, which also uses techniques such as reviews, inspections, walk-through. Validation is the process of checking what has been specified is what the user actually wanted.

* Validation: Are we doing the right job?
* Verification: Are we doing the job right?

In order to achieve consistency in the Testing style, it is imperative to have and follow a set of testing principles. This enhances the efficiency of testing within SQA team members and thus contributes toincreased productivity. The purpose of this document is to provide an overview of the testing, plus the techniques. Here, after training is done on the training dataset, testing is done.

**WHITE BOX TESTING:**

In designing a database the flow of specific inputs through the code, expected output and the functionality of conditional loops are tested.

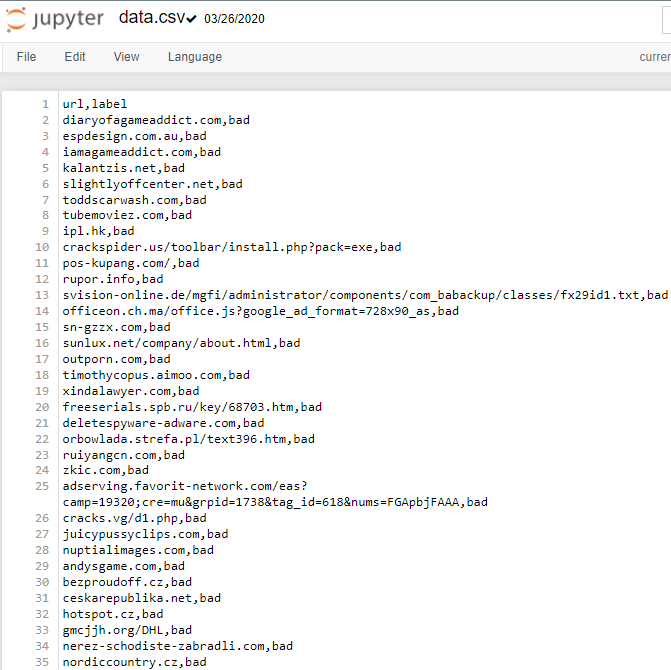
At SDEI, 3 levels of software testing is done at various SDLC phases

1. **UNIT TESTING**: in which each unit (basic component) of the software is tested to verify that the detailed design for the unit has been correctly implemented
2. **INTEGRATION TESTING**: in which progressively larger groups of tested software components corresponding to elements of the architectural design are integrated and tested until the software works as a whole.
3. **SYSTEM TESTING**: in which the software is integrated to the overall product and tested to show that all requirements are met. A further level of testing is also done, in accordance with requirements:
4. **REGRESSION TESTING**: is used to refer the repetition of the earlier successful tests to ensure that changes made in the software have not introduced new bugs/side effects.
5. **ACCEPTANCE TESTING:** Testing to verify a product meets customer specified requirements. The acceptance test suite is run against supplied input data. Then the results obtained are compared with the expected results of the client. A correct match was obtain.

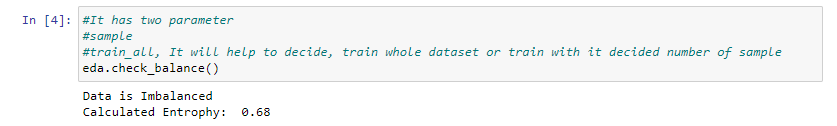
**CHAPTER - 7**

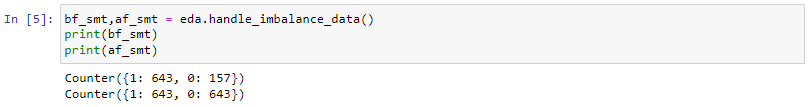
**RESULT**

**1.Gathering the Data**

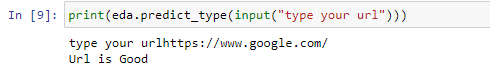
****

1. **Gaussian Naive Bayes:**





**Checking the URL:**



## CHAPTER - 8

## CONCLUSION

This paper aims to enhance detection methods to detect phishing websites using machine learning technology. We achieved 71.13% detection accuracy using Gaussian Naive Bayes algorithm with lowest false positive rate. Also the result shows that classifiers give better performance when we use more data as training data.

### CHAPTER - 9

### FUTURE SCOPE

We will keep working on training the model to get a better accuracy on the predictions and give more datasets so that we can increase the performance .And in the future we can train the model with the latest ML algorithms

### CHAPTER - 10

### BIBLIOGRAPHY

1. R. K. Nepali and Y. Wang, Y., “You look suspicious!! Leveraging visible attributes to classify malicious short urls on twitter”, 2016 49th Hawaii International Conference on System Sciences (HICSS). IEEE, pp. 2648–2655, 2016.
2. N. Sadeh, A. Tomasic, and I Fette, “Learning to detect phishing emails”, Proceedings of the 16th international conference on world wide web, pp.649–656, 2007.
3. https://resources.infosecinstitute.com/category/enterprise/phishing/the-phishing-landscape/phishing-data-attackstatistics/#gref
4. Mahmoud Khonji, Youssef Iraqi, "Phishing Detection: A Literature Survey IEEE, and Andrew Jones, 2013
5. Mohammad R., Thabtah F. McCluskey L., (2015) Phishing websites dataset. Available: https://archive.ics.uci.edu/ml/datasets/Phishing+Websites Accessed January 2016.
6. N. Lord, “What is a Phishing Attack? Defining and Identifying Different

Types of Phishing Attacks”. https://digitalguardian.com/blog/whatphishing-attackdefining-and-identifying-different-types-phishingattacks, 2018.

7. N. Sadeh, A. Tomasic, and I Fette, “Learning to detect phishing emails”, Proceedings of the 16th international conference on world wide web, pp.649–656, 2007.

8. S. Purkait, “Phishing counter measures and their effectiveness–literature review”, Information Management & Computer Security, vol. 20, no. 5, pp. 382–420, 2012.