FEATURE FREE METHOD FOR DETECTING THE PHISHING WEBSITES

A PROJECT REPORT

Submitted by

K.BUVANESHWARI(920119104003)

R.SNEHA(920119104015)

P.SOWMYA(920119104017)

T.VIJI(920119104020)

in partial fulfillment for the award of the degreeof

BACHELOR OF ENGINEERING

COMPUTER SCIENCE AND ENGINEERING

BHARATH NIKETAN ENGINEERING COLLEGE

AUNDIPATTY

ANNA UNIVERSITY : CHENNAI 600 025

MAY 2023

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"FEATURE FREE METHOD FOR DETECTING THE PHISHING WEBSITE"** is the bonafide work of "**BUVANESHWARI.K(920119104003),SNEHA.R(920119104015), SOWMYA.P(920119104017),VIJI.T(90119104020)"**who carried out the project work under my supervision.

SIGNATURE

HEAD OF THE DEPARTMENT

Mr.B.Sankarapandiyan.ME
Department of Computer Science and Engineering
Bharath Niketan Engineering College,Aundipatty-625 536

SIGNATURE

PROJECT SUPERVISOR

Mrs.K.Saravanaselvi.ME
Department of Information Technology
Bharath Niketan Engineering College, Aundipatty-625 536

Submitted for the Project PhaseVivaVoce examination held on .....

ACKNOWLEDGEMENT

We thank the most graceful creator of the universe, our almighty **GOD,** who ideally supported us thought this project.

At this moment of having successfully completed our project, we wish to convey our sincere thanks to the management and our Chairman **Dr.S.MOHAN** who provide all the facilities to us.

We would like to express our sincere thanks to our Principal **Dr.P.V.ARUL KUMAR, M.E, M.B.A, Ph.D.,** for letting us to do our project and offering adequate duration in completing our project.

we are also grateful to **Mr.B.SANKARAPANDIAN,M.E.,** our Head of the Department for his constructive suggestions during our project with deep sense of gratitude.

We extend our sincere thanks to our guide **Mrs.K.SARAVANASELVI M.E.,**Assistant Professor, Department of Information Technology, who is our light house in the vast ocean of learning with her inspiring guideless and encouragement to complete the project.

We would like to express our gratitude to all the teaching and non- teaching staff members of Computer Science and Engineering Department andour friends for that kind help extended to us.

ABSTRACT

The Internet has become an indispensable part of our life, However, It also has provided opportunities to anonymously perform malicious activities like Phishing. Phishers try to deceive their victims by social engineering or creating mock up websites to steal information such as account ID, username, password from individuals and organizations. Phishing is an electronic online identity theft in which the attackers use a combination of social engineering and web site spoofing techniques to trick a user into revealing confidential information. This information is typically used to make an illegal economic profit (e.g., by online banking transactions, purchase of goods using stolen credentials, etc.). - The Internet has a remarkable platform for common people communication. Persons with criminal mind have found a way of stealing personal information without actually meeting them and with the least risk of being caught. It is called Phishing. Phishing poses a huge threat to the e-commerce industry. Not only does it shatter the confidence of customers towards e-commerce, but also causes electronic service providers tremendous economic loss. Hence it is essential to know about phishing. Although many methods have been proposed to detect phishing websites, Phishers have evolved their methods to escape from these detection methods. One of the most successful methods for detecting these malicious activities is Machine Learning. This is because most Phishing attacks have some common characteristics which can be identified by machine learning methods.

TABLE OF CONTENTS

CHAPTER 1

1.INTRODUCTION

Online services simplify our lives. They allow us to access information ubiquitously and are also useful for service providers because they reduce the operational costs involved in offering a service. For example, online banking over the web has become indispensable for customers as well as for banks. Unfortunately, interacting with an online service such as a banking web application often requires a certain degree of technical sophistication that not all Internet users possess. For the last couple of years, such naive users have been increasingly targeted by phishing attacks that are launched by miscreants who are aiming to make an easy profit by means of illegal financial transactions.

Phishing is a form of electronic identity theft in which a combination of social engineering and web site spoofing techniques are used to trick a user into revealing confidential information with economic value. In a typical attack, the attacker sends a large number of spoofed (i.e., fake) e-mails to random Internet users that appear to be coming from a legitimate business organization such as a bank.

The e-mail urges the recipient (i.e., the potential victim) to update his personal information. The site is prepared in a way such that it looks familiar to the victim. That is, the phishers typically imitate the visual corporate identity of the target organization by using similar colors, icons, logos and textual descriptions. In order to “update” his personal information, the victim is asked to enter his online banking login credentials (i.e., user name and password) to access the web site. If a victim enters his valid login credentials into the fraudulent web site, the phisher can then impersonate the victim. This may allow the attacker to transfer funds from the victim’s account or cause

other damage. Because victims are directly interacting with a web site that they believe they know and trust, the success rate of such attacks is very high. significantly over the last years. For example, the number of unique phishing web sites has exploded from 7,197 in December 2005 to 28,531 in December 2006.

We automatically tested the effectiveness of the blacklists maintained by Google and Microsoft over a three week period. During this time, we tested the blacklists with 10,000 phishing URLs. Phishing pages have to be quite similar to the authentic pages in order to deceive users. For this reason, phishers normally use a technique called visual deception. Therefore, it should be possible to detect phishing pages by analysing the visual similarity of a suspect page and the authentic webpage. However, to complicate and evade the detection process, phishers tend to use different representation techniques to create visually similar phishing pages.

We call this kind of countermeasure phishing page polymorphism, after the polymorphism of computer viruses. A polymorphic virus normally contains a variety of different encryption schemes that require different decryption routines. Anti-Phishing Working Group (APWG) emphasizes that phishing attacks have grown in recent years, Since machine learning methods proved to be a powerful tool for detecting patterns in data, these methods have made it possible to detect some of the common phishing traits, therefore, recognizing phishing websites.

In this paper, we provide a comparative and analytical evaluation of different machine learning methods on detecting the phishing websites. The machine learning methods that we studied are Logistic Regression, Decision Tree.

CHAPTER 2

2.LITERATURE SURVEY

Title: STUDY ON PHISHING ATTACKS AND ANTIPHISHING TOOLS

Year: 2016

Author: Dr. Radha Damodaram

Methodology

Phishing is the attempt to acquire sensitive information such as usernames, passwords, and credit card details (and sometimes, indirectly, money), often for malicious International Research Journal of Engineering and Technology reasons, by masquerading as a trustworthy entity in an electronic communication. Now days it has become very serious. There are many techniques to solve these problems. But people may don't aware of the seriousness of phishing. Periodical updating of anti-phishing tools or softwares in their own systems may helpful to secure their confidential information and credentials. This study may give the awareness about the phishing problems and solutions.

Advantage

They have several advantages over parametric classifiers such as discriminate analysis. The advantage of this work is to compare the performance on the basis of its accuracy, time taken to build model, and training data set size.

Disadvantage

The most common problem in the selection of attributes is using fast and accurate algorithm which doesn't require long time to run and give accurate and correct results.

Title: COUNTERACTING PHISHING PAGE POLYMORPHISM:AN IMAGE LAYOUT ANALYSIS APPROACH

Year: 2012

Author: Ieng-Fat Lam, Wei-Cheng Xiao, Szu-Chi Wang, and Kuan-Ta Chen

Methodology

In this paper, we have proposed a polymorphic phishing page detection mechanism based on layout similarity analysis. To cope with polymorphic counterattacks from phishers, we apply image processing techniques and analyze the layout of the page rather than the text content or the HTML codes. The image-based phishing detection mechanism is more robust than the HTML-based approach because it is more adaptable to phishing page polymorphism. In our experiments, 6, 750 phishing pages and 312 authentic pages were analysed and evaluated. The results show that our mechanism achieves an accuracy rate of 99.6%, a false positive rate of less than 0.028%, and a false negative rate of less than 0.003%

Advantage

The aim was to reduce the ratio gap between the majority classes with the minority class. The proposed method is found to be useful for such datasets

where the class labels are not certain and can also help to overcome the problem of phishing datasets and also for other data domains.

Disadvantage

The outcome labels of most of the phishing datasets are not consistent with the underlying data. The conventional over-sampling and under-sampling technique may not always be appropriate for such datasets.

Title: ADVANCED SOCIAL ENGINEERING ATTACKS

Year: 2014

Author: Katharina Krombholz, Heidelinde Hobel, Markus Huber, Edgar Weippl

Methodology

In this paper, we described common attack scenarios for modern social engineering attacks on knowledge workers. BYOD-policies and distributed collaboration as well as communication over third-party channels offers a variety of new attack vectors for advanced social engineering attacks. We believe that a detailed understanding of the attack vectors is required to develop efficient countermeasures and protect knowledge workers from social engineering attacks. To facilitate this, we introduced a comprehensive taxonomy of attacks, classifying them by attack channel, operator, different types of social engineering and specific attack scenarios. We discussed real-world examples and advanced attack vectors used in popular communication channels and the specific issues of computer-supported collaboration of knowledge workers in the business

environment such as cloud services, social networks and mobile devices as part of BYOD policies. In this paper, we not only discussed complex advanced attack scenarios, but also provided a comprehensive classification that can serve as a basis for the development of countermeasures and further inter disciplinary research in the field.

Advantage

In these cases the output took the form of classification rules, which are basic knowledge representation styles that many machine learning methods used.

Disadvantage

The weather problem is a tiny dataset that we will use repeatedly to illustrate machine learning methods.

Title: School of Phish: A Real-World Evaluation of Anti-Phishing Training

Year: 2009

Author: Ponnurangam Kumaraguru, Justin Cranshaw, Alessandro Acquisti,Lorrie Cranor, Jason Hong, Mary Ann Blair, Theodore Pham

Methodology

In this paper, we investigated the effectiveness of an embedded training methodology called PhishGuru that teaches people about phishing during their normal use of email. We showed that, even 28 days after training, users trained by PhishGuru were less likely to click on the link in a simulated phishing email than those who were not trained. Further-more, users who saw the training intervention twice were less likely to give information to fake phishing websites

than those who only saw the training intervention once. Addition-ally, results from this study indicate that training users to recognize phishing emails using Phish Guru does not increase their concern towards email in general or cause them to make more false positive mistakes. Another surprising result was that around 90% of the participants who eventually click edon the link in an email did so within 8 hours of the time the email was sent. We believe this behavior generalizes to other university populations, though non-university populations may behave quite differently when reading emails.

Advantage

The solutions to the class-imbalance problem were previously proposed both at the data and algorithmic levels.

Disadvantage

In particular, they implicitly assume that all misclassification errors cost equally.

Title: ON THE EFFECTIVENESS OF TECHNIQUES TO DETECT PHISHING SITES

Year: 2013

Author: Christian Ludl, Sean McAllister, Engin Kirda, Christopher Kruegel

Methodology

In this paper, we reported our findings on analyzing the effectiveness of two popular anti-phishing solutions. We tested the anti-phishing solutions integrated into the Firefox 2 (i.e., Google blacklists) and Microsoft's Internet

Explorer 7 over a period of three weeks. We fed these blacklists 10,000 phishing URLs to measure their effectiveness in mitigating phishing attacks. Furthermore, by analysing a large number of phishing pages, we report page properties that can be used to identify phishing pages and improve existing solutions. Our findings show that blacklist-based solutions are actually quite effective in protecting users against phishing attempts and that such solutions are an important and useful component in the fight against phishing.

Advantage

This is the ability to defect features is the primary advantage of the mechanism

Disadvantage

The soft computing does not perform much symbolic manipulation.

CHAPTER 3

3.1 EXISTING SYSTEM

- The Existing system uses **Normal Compression Distance** to find similarity between websites, that doesn't classify the data accurately.
- For increasing the accuracy of the data classification and prediction the proposed system is introduced.
- Classification using large volume of data decreases the accuracy of the classification.

3.1.1DISADVANTAGES

- Theoretical Limits.
- Loss of Information.
- Incorrect Classification Results.

3.2 PROPOSED SYSTEM

- Also, we evaluate the **Principal Component Analysis** is an unsupervised learning algorithm that is used for the dimensionality reduction in machine learning. It is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation.
- The **singular value decomposition** (SVD) provides another way to factorize a matrix, into singular vectors and singular values. The SVD is used widely both in the calculation of other matrix operations, such as matrix inverse, but also as a data reduction method in machine learning.

3.2.1ADVANTAGES

- High performance.
- Provide accurate prediction results.
- Reduces the information Loss and the bias of the inference due to the multiple estimates.

CHAPTER 4

4. SYSTEM REQUIREMENTS

4.1 SOFTWARE REQUIREMENTS

- O/S : Windows 7.
- Language : Python
- Front End : Anaconda Navigator - Spyder

4.2 HARDWARE REQUIREMENTS

- System : Pentium IV 2.4 GHz
- Hard Disk : 200 GB
- Mouse : Logitech.
- Keyboard : 110 keys enhanced
- Ram : 4GB

CHAPTER 5

5. MODULES

- Data Selection and Loading
- Data Preprocessing
- Splitting Dataset into Train and Test Data
- Feature Extraction
- Classification
- Prediction
- Result Generation

CHAPTER 6

6.MODULES DESCRIPTION

6.1 DATA SELECTION AND LOADING

- The data selection is the process of selecting the data predicting the phishing attack
- In this project, the phishing dataset is used for predicting phishing attack.
- The dataset which contains the information about Index,UsingIP,LongURL,ShortURL,Symbol@,Redirecting//,PrefixSuffix,SubDomains,HTTPS,DomainRegLen,Favicon,NonStdPort,HTTPSDomainURL,RequestURL,AnchorURL,LinksInScriptTags,ServerFormHandler,InfoEmail,AbnormalURL,WebsiteForwarding,StatusBarCust,DisableRightClick,UsingPopupWindow,IframeRedirection,AgeofDomain,DNSRecording,WebsiteTraffic,PageRank,GoogleIndex,LinksPointingToPage,StatsReport,class.

6.2 DATA PREPROCESSING

- Data pre-processing is the process of removing the unwanted data from the dataset.
    - ❖ Missing data removal
    - ❖ Encoding Categorical data
- Missing data removal: In this process, the null values such as missing values are removed using imputer library.
- Encoding Categorical data: That categorical data is defined as variables with a finite set of label values. That most machine learning algorithms require numerical input and output variables. That an integer and one hot encoding is used to convert categorical data to integer data.

6.3 SPLITTING DATASET INTO TRAIN AND TEST DATA

- Data splitting is the act of partitioning available data into. Two portions, usually for cross-validators purposes.
- One portion of the data is used to develop a predictive model and the other to evaluate the model's performance.
- Separating data into training and testing sets is an important part of evaluating data mining models.
- Typically, when you separate a data set into a training set and testing set, most of the data is used for training, and a smaller portion of the data is used for testing.
- It determines if the association between two categorical variables of the sample would reflect their real association in the populations

6.4 FEATURE EXTRACTION

- Feature Scaling or Standardization: It is a step of Data Pre Processing which is applied to independent variables or features of data. It basically helps to normalise the data within a particular range. Sometimes, it also helps in speeding up the calculations in an algorithm.
- **Principal Component Analysis** is an unsupervised learning algorithm that is used for the dimensionality reduction in machine learning. It is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation.
- The **singular value decomposition** (SVD) provides another way to factorize a matrix, into singular vectors and singular values. The SVD is used widely both in the calculation of other matrix operations, such as matrix inverse, but also as a data reduction method in machine learning.

5.5 CLASSIFICATION

- The Supervised classification algorithm such as Support vector machine, Decision tree, Logistic Regression is used, Unsupervised learning such as K-Nearest Neighbour in Data Mining
- **Random forests** or random decision forests are an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees.
- **K nearest neighbours** is a simple *algorithm* that stores all available cases and classifies new cases based on a similarity measure (e.g., distance functions). *KNN* has been used in statistical estimation and pattern recognition and non-parametric technique.

5.6 PREDICTION

- It's a process of predicting phishing from the dataset.
- This project will effectively predict the data from dataset by enhancing the performance of the overall prediction results.

5.7 RESULT GENERATION

The Final Result will get generated based on the overall classification and prediction. The performance of this proposed approach is evaluated using some measures like,

- Accuracy
- Graph based on prediction

CHAPTER 7

7.DIAGRAMS

7.1ARCHITECTURE DIAGRAM

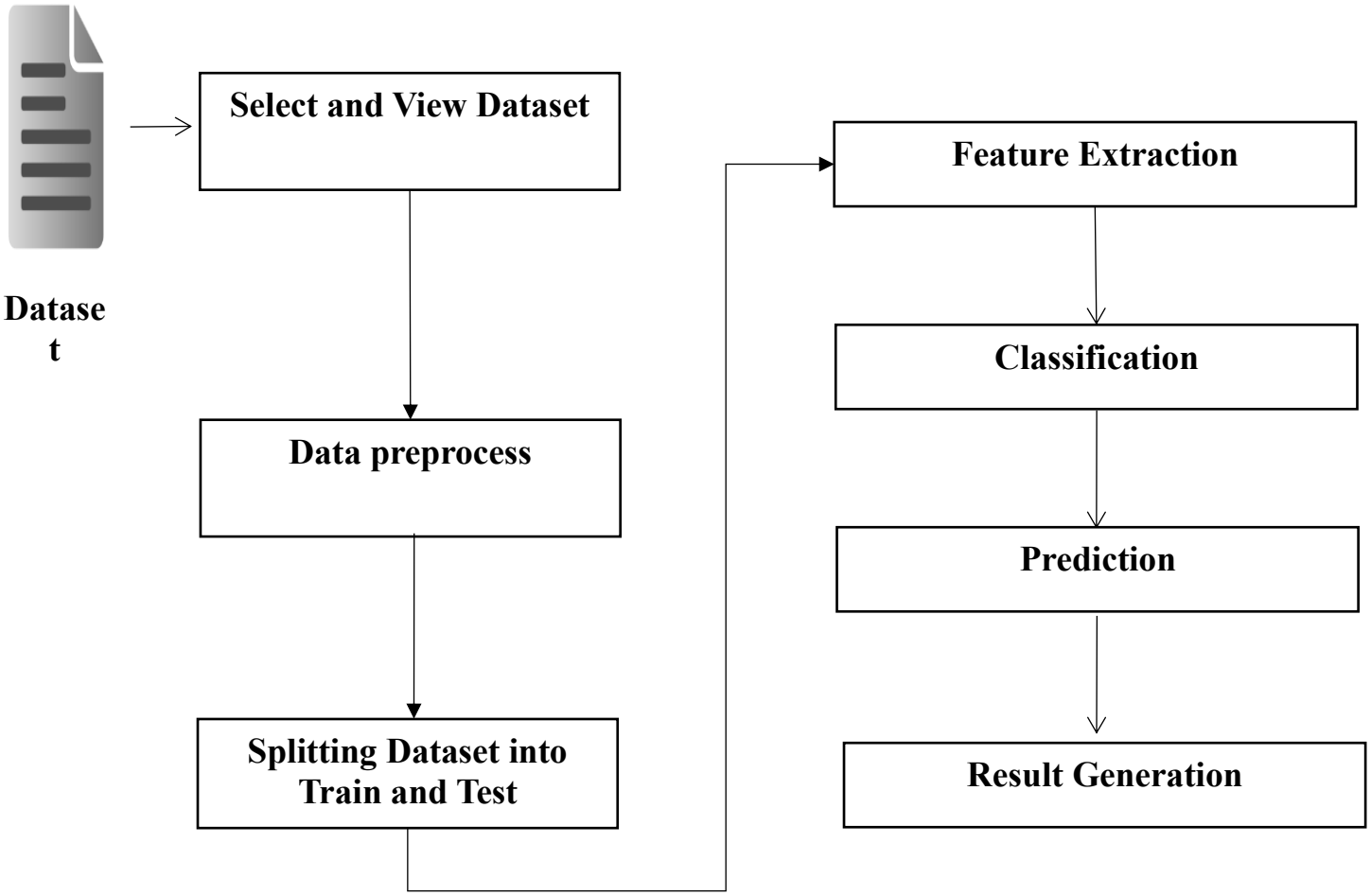


Fig 7.1 ARCHITECTURE DIAGRAM

7.2FLOW DIAGRAM

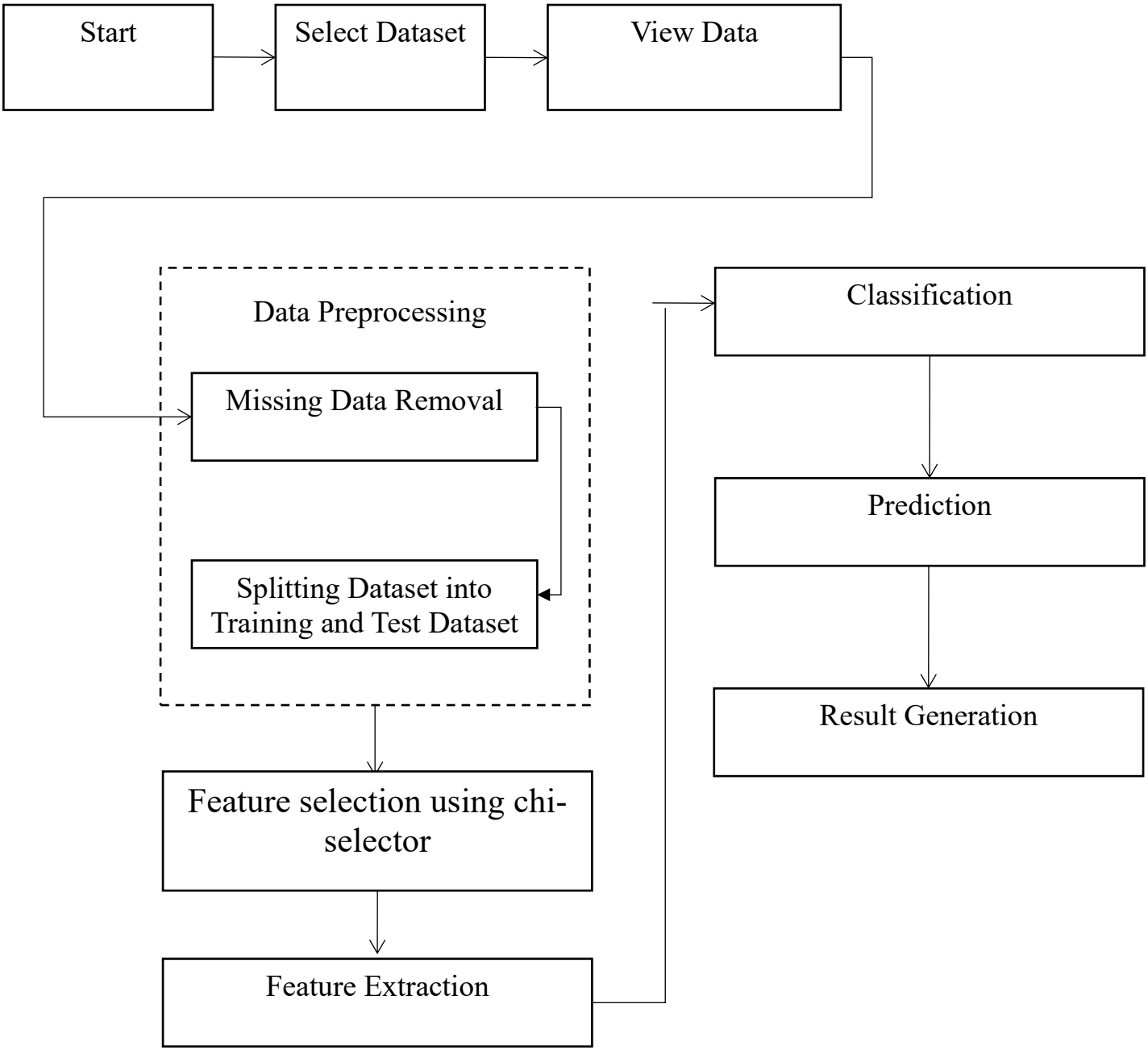


Fig 7.2FLOW DIAGRAM

7.3CLASS DIAGRAM

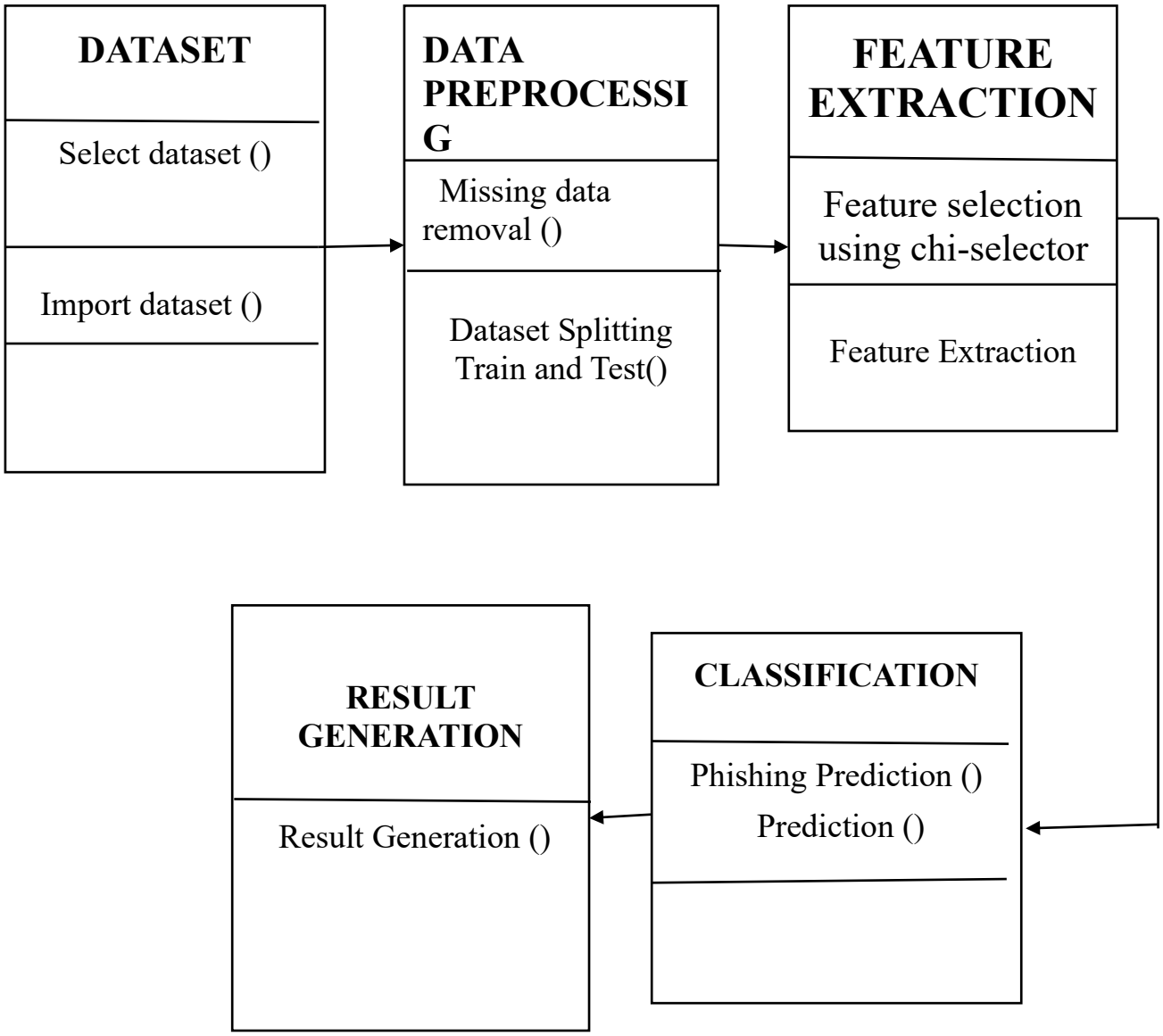


Fig 7.3 CLASS DIAGRAM

7.3 SEQUENCE DIAGRAM

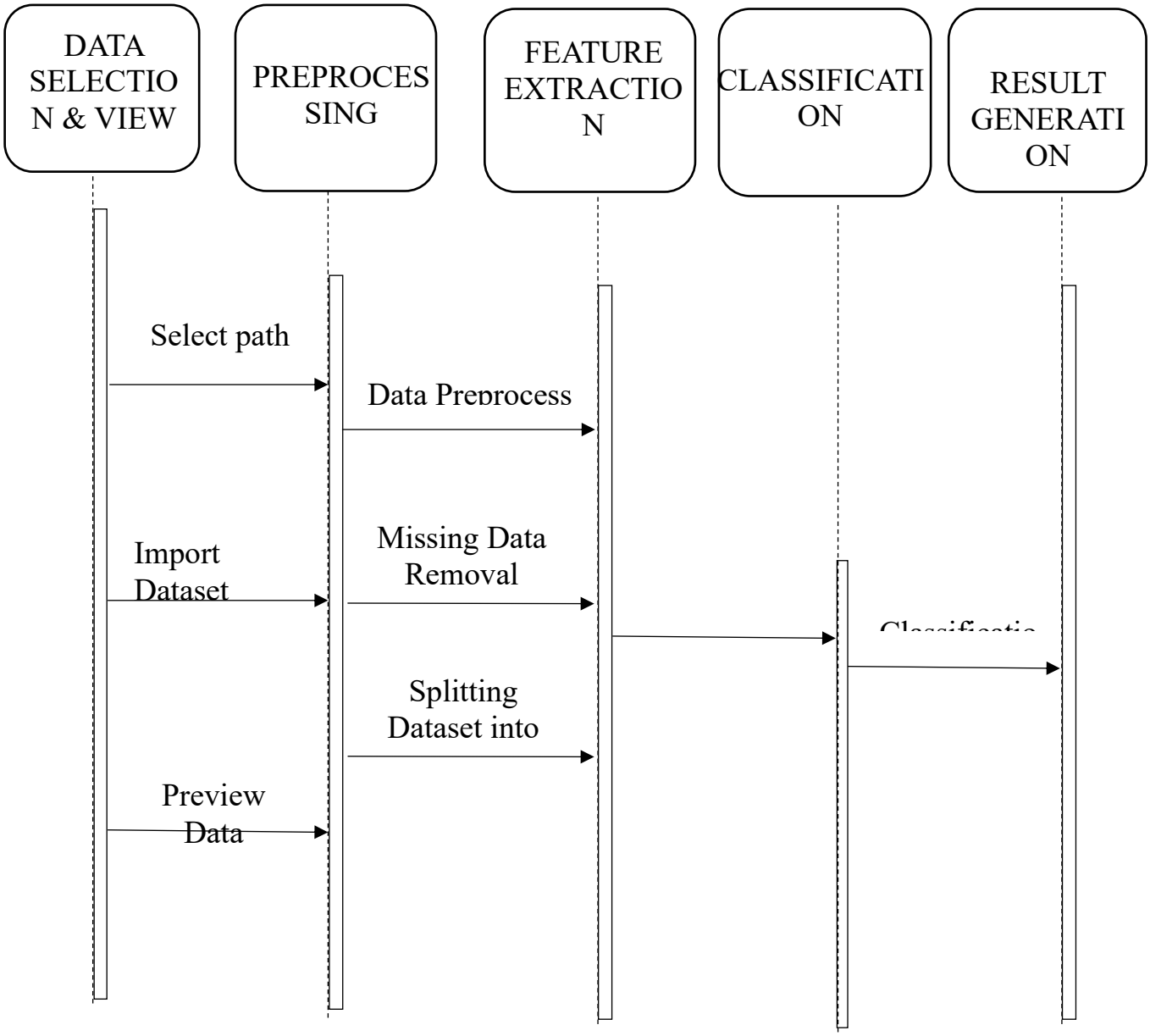


Fig 6.2.2SEQUENCE DIAGRAM

CHAPTER 8

8.LIBRARIES DETAILS

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

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

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

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

Extensions or modules for SciPy care conventionally named SciKits. As such, the module provides learning algorithms and is named scikit-learn.
The vision for the library is a level of robustness and support required for use in production systems. This means a deep focus on concerns such as easy of use, code quality, collaboration, documentation and performance.

The library is focused on modeling data. It is not focused on loading, manipulating and summarizing data. For these features, refer to NumPy and Pandas.

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

- Clustering: for grouping unlabeled data such as KMeans.
- Cross Validation: for estimating the performance of supervised models on unseen data.
- Datasets: for test datasets and for generating datasets with specific properties for investigating model behavior.
- Dimensionality Reduction: for reducing the number of attributes in data for summarization, visualization and feature selection such as Principal component analysis.
- 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 generalized linear models, discriminate analysis, naive bayes, lazy methods, neural networks, support vector machines and decision trees.

8.1 PANDA

- A fast and efficient DataFrame object for data manipulation with integrated indexing;
- Tools for reading and writing data between in-memory data structures and different formats: CSV and text files, Microsoft Excel, SQL databases, and the fast HDF5 format;

- Intelligent data alignment and integrated handling of missing data: gain automatic label-based alignment in computations and easily manipulate messy data into an orderly form;
- Flexible reshaping and pivoting of data sets;
- Intelligent label-based slicing, fancy indexing, and subsetting of large data sets;
- Columns can be inserted and deleted from data structures for size mutability;
- Aggregating or transforming data with a powerful group by engine allowing split-apply-combine operations on data sets;
- High performance merging and joining of data sets;
- Hierarchical axis indexing provides an intuitive way of working with high-dimensional data in a lower-dimensional data structure;
- Time series-functionality: date range generation and frequency conversion, moving window statistics, moving window linear regressions, date shifting and lagging. Even create domain-specific time offsets and join time series without losing data;
- Highly optimized for performance, with critical code paths written in Cythonor C.
- Python with *pandas* is in use in a wide variety of academic and commercial domains, including Finance, Neuroscience, Economics, Statistics, Advertising, Web Analytics, and more.

8.2MATLIBPLOTTING

- 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
- At first sight, it will seem that there are quite some components to consider when you start plotting with this Python data visualization library. You'll probably agree with me that it's confusing and sometimes even discouraging seeing the amount of code that is necessary for some plots, not knowing where to start yourself and which components you should use.
- Luckily, this library is very flexible and has a lot of handy, built-in defaults that will help you out tremendously. As such, you don't need much to get started: you need to make the necessary imports, prepare some data, and you can start plotting with the help of the plot() function! When you're ready, don't forget to show your plot using the show() function.

CHAPTER 9

9.SOFTWARE DESCRIPTION

9.1 PYTHON

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). This tutorial gives enough understanding on Python programming language.

Python is a popular programming language. It was created in 1991 by Guido van Rossum.

It is used for:

- web development (server-side),
- software development,
- mathematics,
- System scripting.

Python can be used on a server to create web applications. Python can be used alongside software to create workflows. Python can connect to database systems. It can also read and modify files. Python can be used to handle big data and perform complex mathematics. Python can be used for rapid prototyping, or for production-ready software development.

Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc). Python has a simple syntax similar to the English language. Python has syntax

that allows developers to write programs with fewer lines than some other programming languages.

Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick. Python can be treated in a procedural way, an object-orientated way or a functional way.

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

9.1.1 HISTORY OF PYTHON

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

9.1.2 PYTHON FEATURES

Python's features include −

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** − you can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** − Python provides interfaces to all major commercial databases.
- **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Python is available on a wide variety of platforms including Linux and Mac OS X.

Python Syntax compared to other programming languages

- Python was designed to for readability, and has some similarities to the English language with influence from mathematics.
- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
- Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

9.1.3 FEATURES OF PYTHON

Simple

Python is a simple and minimalistic language. Reading a good Python program feels almost like reading English, although very strict English! This pseudo-code nature of Python is one of its greatest strengths. It allows you to concentrate on the solution to the problem rather than the language itself.

Easy to Learn

As you will see, Python is extremely easy to get started with. Python has an extraordinarily simple syntax, as already mentioned.

Free and Open Source

Python is an example of a *FLOSS* (Free/Libré and Open Source Software). In simple terms, you can freely distribute copies of this software, read its source code, make changes to it, and use pieces of it in new free programs. FLOSS is based on the concept of a community which shares knowledge. This is one of the reasons why Python is so good - it has been created and is constantly improved by a community who just want to see a better Python.

High-level Language

When you write programs in Python, you never need to bother about the low-level details such as managing the memory used by your program, etc.

Portable

Due to its open-source nature, Python has been ported to (i.e. changed to make it work on) many platforms. All your Python programs can work on any of these platforms without requiring any changes at all if you are careful enough to avoid any system-dependent features.

You can use Python on GNU/Linux, Windows, FreeBSD, Macintosh, Solaris, OS/2, Amiga, AROS, AS/400, BeOS, OS/390, z/OS, Palm OS, QNX, VMS, Psion, Acorn RISC OS, VxWorks, PlayStation, Sharp Zaurus, Windows CE and PocketPC!

You can even use a platform like Kivy to create games for your computer and for iPhone, iPad, and Android.

Interpreted

This requires a bit of explanation.

A program written in a compiled language like C or C++ is converted from the source language i.e. C or C++ into a language that is spoken by your computer (binary code i.e. 0s and 1s) using a compiler with various flags and options. When you run the program, the linker/loader software copies the program from hard disk to memory and starts running it.

Python, on the other hand, does not need compilation to binary. You just run the program directly from the source code. Internally, Python converts the source code into an intermediate form called bytecodes and then translates this into the native language of your computer and then runs it. All this, actually, makes using Python much easier since you don't have to worry about compiling the program, making sure that the proper libraries are linked and loaded, etc. This also makes your Python programs much more portable, since you can just copy your Python program onto another computer and it just works!

Object Oriented

Python supports procedure-oriented programming as well as object-oriented programming. In *procedure-oriented* languages, the program is built around procedures or functions which are nothing but reusable pieces of programs. In object-oriented languages, the program is built around objects which combine data and functionality. Python has a very powerful but simplistic way of doing OOP, especially when compared to big languages like C++ or Java.

Extensible

If you need a critical piece of code to run very fast or want to have some piece of algorithm not to be open, you can code that part of your program in C or C++ and then use it from your Python program.

You can embed Python within your C/C++ programs to give scripting capabilities for your program's users.

Extensive Libraries

The Python Standard Library is huge indeed. It can help you do various things involving regular expressions, documentation generation, unit testing, threading, databases, web browsers, CGI, FTP, email, XML, XML-RPC, HTML, WAV files, cryptography, GUI (graphical user interfaces), and other system-dependent stuff. Remember, all this is always available wherever Python is installed. This is called the *Batteries Included* philosophy of Python.

Besides the standard library, there are various other high-quality libraries which you can find at the Python Package Index.

Summary

Python is indeed an exciting and powerful language. It has the right combination of performance and features that make writing programs in Python both fun and easy.

Python 3 versus 2

You can ignore this section if you're not interested in the difference between "Python version 2" and "Python version 3". But please do be aware of which version you are using. This book is written for Python version 3.

Remember that once you have properly understood and learn to use one version, you can easily learn the differences and use the other one. The hard part is learning programming and understanding the basics of Python language itself. That is our goal in this book, and once you have achieved that goal, you can easily use Python 2 or Python 3 depending on your situation.

What makes Python so special? How does it happen that programmers, young and old, experienced and novice, want to use it? How did it happen that large companies adopted Python and implemented their flagship products using it?

There are many reasons – we've listed some of them already, but let's enumerate them again in a more practical manner:

- It is **easy to learn** – the time needed to learn Python is shorter than for many other languages; this means that it's possible to start the actual programming faster;

- It is **easy to use** for writing new software – it's often possible to write code faster when using Python;
- It is **easy to understand** – it's also often easier to understand someone else's code faster if it is written in Python;
- It is **easy to obtain**, install and deploy – Python is free, open and multiplatform; not all languages can boast that.

9.2 ANACONDA

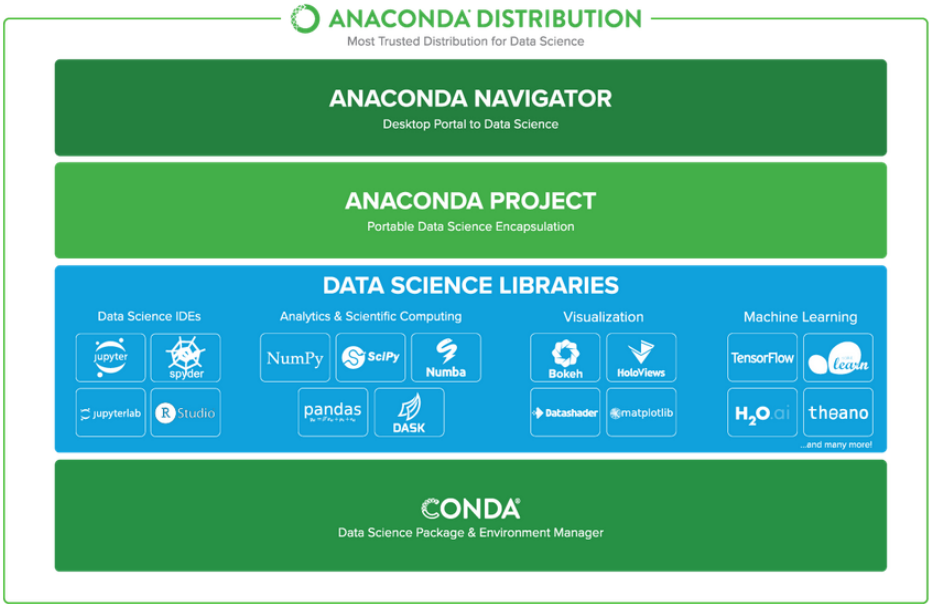
Anaconda is the most popular python data science platform.

9.2.1 ANACONDA DISTRIBUTION

With over 6 million users, the open source Anaconda Distribution is the fastest and easiest way to do Python and R data science and machine learning on Linux, Windows, and Mac OS X. It's the industry standard for developing, testing, and training on a single machine.

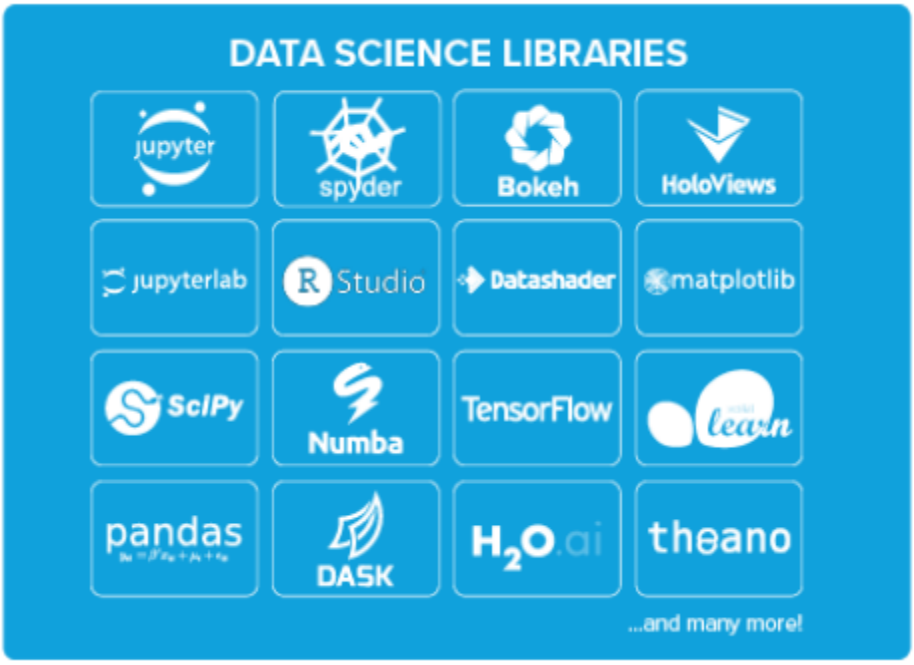
9.2.2 ANACONDA ENTERPRISE

Anaconda Enterprise is an AI/ML enablement platform that empowers organizations to develop, govern, and automate AI/ML and data science from laptop through training to production. It lets organizations scale from individual data scientists to collaborative teams of thousands, and to go from a single server to thousands of nodes for model training and deployment.



9.2.3Anaconda Data Science Libraries

- Over 1,400 Anaconda-curated and community data science packages
- Develop data science projects using your favourite IDEs, including Jupyter, JupyterLab, Spyder, and RStudio
- Analyse data with scalability and performance with Dask, numpy, pandas, and Numba
- Visualize your data with Matplotlib, Bokeh, Datashader, and Holoviews
- Create machine learning and deep learning models with Scikit-learn, Tensorflow, h20, and Theano

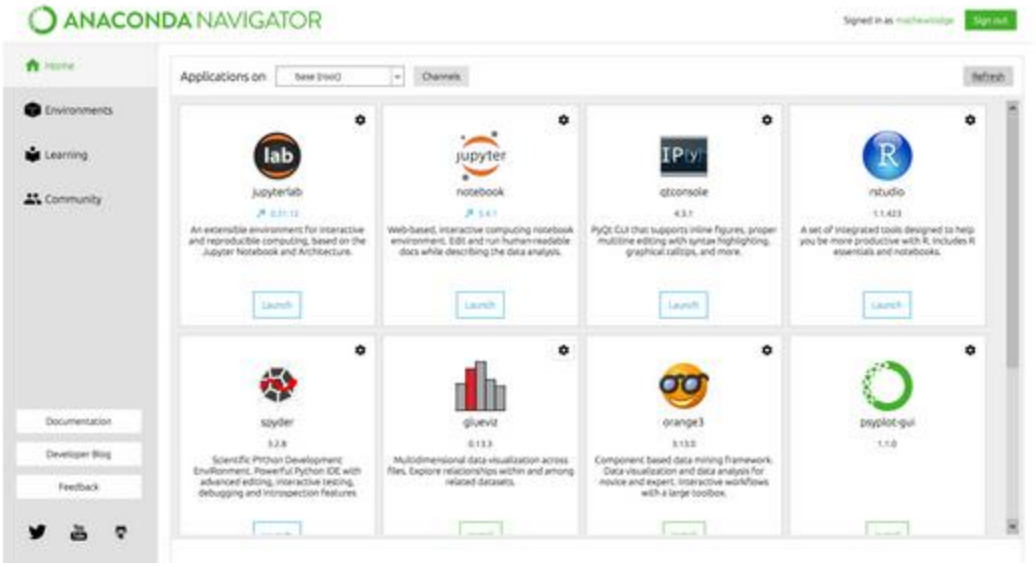


Conda, the Data Science Package & Environment Manager

- Automatically manages all packages, including cross-language dependencies
- Works across all platforms: Linux, macOS, Windows
- Create virtual environments
- Download conda packages from Anaconda, Anaconda Enterprise, Conda Forge, and Anaconda Cloud

Anaconda Navigator, the Desktop Portal to Data Science

- Install and launch applications and editors including Jupyter, RStudio, Visual Studio Code, and Spyder
- Manage your local environments and data science projects from a graphical interface
- Connect to Anaconda Cloud or Anaconda Enterprise
- Access the latest learning and community resources



9.3 SPYDER

Spyder is an open source cross-platform integrated development environment (IDE) for scientific programming in the Python language. ... Initially created and developed by Pierre Raybaut in 2009, since 2012 Spyder has been maintained and continuously improved by a team of scientific Python developers and the community. Strongly recommend the free, open-source Spyder Integrated Development Environment (IDE) for scientific and engineering programming, due to its integrated editor, interpreter console, and debugging tools. Spyder is included in Anaconda and other distributions.

Spyder is a powerful scientific environment written in Python, for Python, and designed by and for scientists, engineers and data analysts. It offers a unique combination of the advanced editing, analysis, debugging, and profiling functionality of a comprehensive development tool with the data exploration, interactive execution, deep inspection, and beautiful visualization capabilities of a scientific package.

Beyond its many built-in features, its abilities can be extended even further via its plugin system and API. Furthermore, Spyder can also be used as a PyQt5 extension library, allowing developers to build upon its functionality and embed its components, such as the interactive console, in their own PyQt software.

Editor

Work efficiently in a multi-language editor with a function/class browser, code analysis tools, automatic code completion, horizontal/vertical splitting, and go-to-definition.

IPython Console

Harness the power of as many IPython consoles as you like within the flexibility of a full GUI interface; run your code by line, cell, or file; and render plots right inline.

Variable Explorer

Interact with and modify variables on the fly: plot a histogram or time series, edit a date frame or Numpy array, sort a collection, dig into nested objects, and more!

CHAPTER 10

10. FEASIBILITY STUDY

The feasibility study is carried out to test whether the proposed system is worth being implemented. The proposed system will be selected if it is best enough in meeting the performance requirements.

The feasibility carried out mainly in three sections namely.

- Economic Feasibility
- Technical Feasibility
- Behavioural Feasibility

10.1 ECONOMIC FEASIBILITY

Economic analysis is the most frequently used method for evaluating effectiveness of the proposed system. More commonly known as cost benefit analysis. This procedure determines the benefits and saving that are expected from the system of the proposed system. The hardware in system department if sufficient for system development.

10.2 TECHNICAL FEASIBILITY

This study centre around the system's department hardware, software and to what extend it can support the proposed system department is having the required hardware and software there is no question of increasing the cost of implementing the proposed system. The criteria, the proposed system is technically feasible and the proposed system can be developed with the existing facility.

10.3 BEHAVIOURAL FEASIBILITY

People are inherently resistant to change and need sufficient amount of training, which would result in lot of expenditure for the organization. The proposed system can generate reports with day-to-day information immediately at the user's request, instead of getting a report, which doesn't contain much detail.

CHAPTER 11

11.TESTING OF PRODUCT

11.1 TESTING OF PRODUCT

System testing is the stage of implementation, which aimed at ensuring that system works accurately and efficiently before the live operation commence. Testing 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 error. A successful test is one that answers a yet undiscovered error.

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. The candidate system is subject to variety of tests-on-line response, Volume Street, recovery and security and usability test. A series of tests are performed before the system is ready for the user acceptance testing. Any engineered product can be tested in one of the following ways. Knowing the specified function that a product has been designed to from, test can be conducted to demonstrate each function is fully operational. Knowing the internal working of a product, tests can be conducted to ensure that “al gears mesh”, that is the internal operation of the product performs according to the specification and all internal components have been adequately exercised.

11.1.1 UNIT TESTING

Unit testing is the testing of each module and the integration of the overall system is done. Unit testing becomes verification efforts on the smallest unit of software design in the module. This is also known as ‘module testing’. The modules of the system are tested separately. This testing is carried out during the programming itself. In this testing step, each model is found to be working satisfactorily as regard to the expected output from the module. There are some

validation checks for the fields. For example, the validation check is done for verifying the data given by the user where both format and validity of the data entered is included. It is very easy to find error and debug the system.

11.1.2 INTEGRATION TESTING

Data can be lost across an interface, one module can have an adverse effect on the other sub function, when combined, may not produce the desired major function. Integrated testing is systematic testing that can be done with sample data. The need for the integrated test is to find the overall system performance. There are two types of integration testing. They are:

i) Top-down integration testing.
ii) Bottom-up integration testing.

11.1.3 WHITE BOX TESTING

White Box testing is a test case design method that uses the control structure of the procedural design to drive cases. Using the white box testing methods, we derived test cases that guarantee that all independent paths within a module have been exercised at least once.

11.1.4 BLACK BOX TESTING

- ✓ Black box testing is done to find incorrect or missing function
- ✓ Interface error
- ✓ Errors in external database access
- ✓ Performance errors
- ✓ Initialization and termination errors

In 'functional testing', is performed to validate an application conforms to its specifications of correctly performs all its required functions. So this testing is

also called ‘black box testing’. It tests the external behaviour of the system. Here the engineered product can be tested knowing the specified function that a product has been designed to perform, tests can be conducted to demonstrate that each function is fully operational.

11.1.5 VALIDATION TESTING

After the culmination of black box testing, software is completed assembly as a package, interfacing errors have been uncovered and corrected and final series of software validation tests begin validation testing can be defined as many, but a single definition is that validation succeeds when the software functions in a manner that can be reasonably expected by the customer.

11.1.6 USER ACCEPTANCE TESTING

User acceptance of the system is the key factor for the success of the system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system at the time of developing changes whenever required.

11.1.7 OUTPUT TESTING

After performing the validation testing, the next step is output asking the user about the format required testing of the proposed system, since no system could be useful if it does not produce the required output in the specific format. The output displayed or generated by the system under consideration. Here the output format is considered in two ways. One is screen and the other is printed format. The output format on the screen is found to be correct as the format was designed in the system phase according to the user

needs. For the hard copy also output comes out as the specified requirements by the user. Hence the output testing does not result in any connection in the system.

11.2 TYPES OF SOFTWARE TESTING

AD-HOC TESTING

This type of software testing is very informal and unstructured and can be performed by any stakeholder with no reference to any test case or test design documents. The person performing Ad-hoc testing has a good understanding of the domain and workflows of the application to try to find defects and break the software. Ad-hoc testing is intended to find defects that were not found by existing test cases.

ACCEPTANCE TESTING

Acceptance testing is a formal type of software testing that is performed by end user when the features have been delivered by developers. The aim of this testing is to check if the software confirms to their business needs and to the requirements provided earlier. Acceptance tests are normally documented at the beginning of the sprint (in agile) and is a means for testers and developers to work towards a common understanding and shared business domain knowledge.

ACCESSIBILITY TESTING

In accessibility testing, the aim of the testing is to determine if the contents of the website can be easily accessed by disable people. Various checks such as color and contrast (for color blind people), font size for visually impaired, clear and concise text that is easy to read and understand.

AGILE TESTING

Agile Testing is a type of software testing that accommodates agile software development approach and practices. In an Agile development environment, testing is an integral part of software development and is done along with coding. Agile testing allows incremental and iterative coding and testing.

API TESTING

API testing is a type of testing that is similar to unit testing. Each of the Software APIs are tested as per API specification. API testing is mostly done by testing team unless APIs to be tested or complex and needs extensive coding. API testing requires understanding both API functionality and possessing good coding skills.

AUTOMATED TESTING

This is a testing approach that makes use of testing tools and/or programming to run the test cases using software or custom developed test utilities. Most of the automated tools provided capture and playback facility, however there are tools that require writing extensive scripting or programming to automate test cases.

ALL PAIRS TESTING

Also known as Pair wise testing, is a black box testing approach and a testing method where in for each input is tested in pairs of inputs, which helps to test software works as expected with all possible input combinations.

BETA TESTING

This is a formal type of software testing that is carried out by end customers before releasing or handing over software to end users. Successful completion of Beta testing means customer acceptance of the software.

BLACK BOX TESTING

Black box testing is a software testing method where in testers are not required to know coding or internal structure of the software. Black box testing method relies on testing software with various inputs and validating results against expected output.

BACKWARD COMPATIBILITY TESTING

Type of software testing performed to check newer version of the software can work successfully installed over previous version of the software and newer version of the software works as fine with table structure, data structures, files that were created by previous version of the software.

BOUNDARY VALUE TESTING (BVT)

Boundary Value Testing is a testing technique that is based on concept “error aggregates at boundaries”. In this testing technique, testing is done extensively to check for defects at boundary conditions. If a field accepts value 1 to 100 then testing is done for values 0, 1, 2, 99, 100 and 101.

BIG BANG INTEGRATION TESTING

This is one of the integration testing approaches, in Big Bang integration testing all or all most all of the modules are developed and then coupled together.

BOTTOM UP INTEGRATION TESTING

Bottom up integration testing is an integration testing approach where in testing starts with smaller pieces or sub systems of the software till all the way up covering entire software system. Bottom up integration testing begins with smaller portion of the software and eventually scale up in terms of size, complexity and completeness.

BRANCH TESTING

Is a white box testing method for designing test cases to test code for every branching condition? Branch testing method is applied during unit testing.

BROWSER COMPATIBILITY TESTING

It is one of the sub types of testing of compatibility testing performed by testing team. Browser compatibility testing is performed for web applications with combination of different browsers and operating systems.

COMPATIBILITY TESTING

Compatibility testing is one of the test types performed by testing team. Compatibility testing checks if the software can be run on different hardware, operating system, bandwidth, databases, web servers, application servers, hardware peripherals, emulators, different configuration, processor, different browsers and different versions of the browsers etc.

COMPONENT TESTING

This type of software testing is performed by developers. Component testing is carried out after completing unit testing. Component testing involves testing a group of units as code together as a whole rather than testing individual functions, methods.

CONDITION COVERAGE TESTING

Condition coverage testing is a testing technique used during unit testing, where in developer tests for all the condition statements like if, if else, case etc., in the code being unit tested.

DYNAMIC TESTING

Testing can be performed as Static Testing and Dynamic testing, Dynamic testing is a testing approach where-in testing can be done only by executing code or software are classified as Dynamic Testing. Unit testing, Functional testing, regression testing, performance testing etc.

DECISION COVERAGE TESTING

Is a testing technique that is used in Unit testing, objective of decision coverage testing is to expertise and validate each and every decisions made in the code e.g. if, if else, case statements.

END-TO-END TESTING

End to end testing is performed by testing team, focus of end to end testing is to test end to end flows e.g. right from order creation till reporting or order creation till item return etc. and checking. End to end testing is usually focused mimicking real life scenarios and usage. End to end testing involves testing information flow across applications.

EXPLORATORY TESTING

Exploratory testing is an informal type of testing conducted to learn the software at the same time looking for errors or application behaviour that seems non-obvious. Exploratory testing is usually done by testers but can be done by other stake holders as well like Business Analysts, developers, end users etc. who are interested in learning functions of the software and at the same time looking for errors or behaviour is seems non-obvious.

EQUIVALENCE PARTITIONING

Equivalence partitioning is also known as Equivalence Class Partitioning is a software testing technique and not a type of testing by itself. Equivalence partitioning technique is used in black box and grey box testing types. Equivalence partitioning classifies test data into Equivalence classes as positive Equivalence classes and negative Equivalence classes, such classification ensures both positive and negative conditions are tested.

FUNCTIONAL TESTING

Functional testing is a formal type of testing performed by testers. Functional testing focuses on testing software against design document, Use cases and requirements document. Functional testing is a black box type of testing and does not require internal working of the software unlike white box testing.

FUZZ TESTING

Fuzz testing or fuzzing is a software testing technique that involves testing with unexpected or random inputs. Software is monitored for failures or error messages that are presented due to the input errors.

11.2.2 GUI (GRAPHICAL USER INTERFACE) TESTING

This type of software testing is aimed at testing the software GUI (Graphical User Interface) of the software meets the requirements as mentioned in the GUI mock-ups and Detailed designed documents. For e.g. checking the length and capacity of the input fields provided on the form, type of input field provided, e.g. some of the form fields can be displayed as dropdown box or a set of radio buttons. So GUI testing ensures GUI elements of the software are as per approved GUI mock-ups, detailed design documents and functional requirements. Most of the functional test automation tools work on GUI capture and playback capabilities. This makes script recording faster at the same time increases the effort on script maintenance.

GLASS BOX TESTING

Glass box testing is another name for White box testing. Glass box testing is a testing method that involves testing individual statements, functions etc., Unit testing is one of the Glass box testing methods.

GORILLA TESTING

This type of software testing is done by software testing team, has a scary name though? Objective of Gorilla Testing is to exercise one or few functionality thoroughly or exhaustively by having multiple people test the same functionality.

HAPPY PATH TESTING

Also known as Golden path testing, this type of testing focuses on selective execution of tests that do not exercise the software for negative or error conditions.

INTEGRATION TESTING

Integration testing also known as met in short, in one of the important types of software testing. Once the individual units or components are tested by developers as working then testing team will run tests that will test the connectivity among these units/component or multiple units/components. There are different approaches for Integration testing namely, Top-down integration testing, Bottom-up integration testing and a combination of these two known as Sand witch testing.

INTERFACE TESTING

Software provides support for one or more interfaces like “Graphical user interface”, “Command Line Interface” or “Application programming interface” to interact with its users or other software. Interfaces serves as medium for software to accept input from user and provide result. Approach for interface testing depends on the type of the interface being testing like GUI or API or CLI.

INTERNATIONALIZATION TESTING

Internationalization testing is a type of testing that is performed by software testing team to check the extent to which software can support Internationalization i.e., usage of different languages, different character sets, double byte characters etc., For e.g.: Gmail, is a web application that is used by people all over work with different languages, single by or multi byte character sets.

KEYWORD-DRIVEN TESTING

Keyword driver testing is more of an automated software testing approach than a type of testing itself. Keyword driven testing is known as action driven testing or table driven testing.

LOAD TESTING

Load testing is a type of non-functional testing; load testing is done to check the behaviour of the software under normal and over peak load conditions. Load testing is usually performed using automated testing tools. Load testing intends to find bottlenecks or issues that prevent software from performing as intended at its peak workloads.

LOCALIZATION TESTING

Localization testing a type of software testing performed by software testers, in this type of testing, software is expected to adapt to a particular locale, it should support a particular locale/language in terms of display, accepting input in that particular locale, display, font, date time, currency etc., related to a particular locale. For e.g. many web applications allow choice of locale like English, French, German or Japanese. So once locale is defined or set in the configuration of software, software is expected to work as expected with a set language/locale.

NEGATIVE TESTING

This type of software testing approach, which calls out the “attitude to break”, these are functional and non-functional tests that are intended to break the software by entering incorrect data like incorrect date, time or string or upload binary file when text files supposed to be upload or enter huge text string for input fields etc. It is also a positive test for an error condition.

NON-FUNCTIONAL TESTING

Software are built to fulfil functional and non-functional requirements, non-functional requirements like performance, usability, localization etc., There are many types of testing like compatibility testing, compliance testing, localization testing, usability testing, volume testing etc., that are carried out for checking non-functional requirements.

PAIR TESTING

It is a software testing technique that can be done by software testers, developers or Business analysts (BA). As the name suggests, two people are paired together, one to test and other to monitor and record test results. Pair testing can also be performed in combination of tester-developer, tester-business analyst or developer-business analyst combination. Combining testers and developers in pair testing helps to detect defects faster, identify root cause, fix and test the fix.

PERFORMANCE TESTING

It is a type of software testing and part of performance engineering that is performed to check some of the quality attributes of software like Stability, reliability, availability. Performance testing is carried out by performance engineering team. Unlike Functional testing, Performance testing is done to check non-functional requirements. Performance testing checks how well software

works in anticipated and peak workloads. There are different variations or sub types of performance like load testing, stress testing, volume testing, soak testing and configuration testing.

PENETRATION TESTING

It is a type of security testing, also known as pen test in short. Penetration testing is done to tests how secure software and its environments (Hardware, Operating system and network) are when subject to attack by an external or internal intruder. Intruder can be a human/hacker or malicious programs. Pen test uses methods to forcibly intrude (by brute force attack) or by using a weakness (vulnerability) to gain access to a software or data or hardware with an intent to expose ways to steal, manipulate or corrupt data, software files or configuration. Penetration Testing is a way of ethical hacking, an experienced Penetration tester will use the same methods and tools that a hacker would use but the intention of Penetration tester is to identify vulnerability and get them fixed before a real hacker or malicious program exploits it.

REGRESSION TESTING

It is a type of software testing that is carried out by software testers as functional regression tests and developers as Unit regression tests. Objective of regression tests are to find defects that got introduced to defect fix (is) or introduction of new feature(s). Regression tests are ideal candidate for automation.

RETESTING

It is a type of retesting that is carried out by software testers as a part of defect fix verification.

RISK BASED TESTING

It is a type of software testing and a different approach towards testing a software. In Risk based testing, requirements and functionality of software to be tested are prioritized as Critical, High, Medium and low. In this approach, all critical and high priority tests are tested and them followed by Medium. Low priority or low risk functionality are tested at the end or may not base on the time available for testing.

SMOKE TESTING

It is a type of testing that is carried out by software testers to check if the new build provided by development team is stable enough i.e., major functionality is working as expected in order to carry out further or detailed testing. Smoke testing is intended to find "show stopper" defects that can prevent testers from testing the application in detail. Smoke testing carried out for a build is also known as build verification test.

SECURITY TESTING

It is a type of software testing carried out by specialized team of software testers. Objective of security testing is to secure the software is to external or internal threats from humans and malicious programs. Security testing basically checks, how good is software's authorization mechanism, how strong is authentication, how software maintains confidentiality of the data, how does the software maintain integrity of the data, what is the availability of the software in an event of an attack on the software by hackers and malicious programs is for Security testing requires good knowledge of application, technology, networking, security testing tools. With increasing number of web applications necessarily of security testing has increased to a greater extent.

SANITY TESTING

It is a type of testing that is carried out mostly by testers and in some projects by developers as well. Sanity testing is a quick evaluation of the software, environment, network, external systems are up & running, software environmentas a whole is stable enough to proceed with extensive testing. Sanity tests are narrow and most of the time sanity tests are not documented.

SCALABILITY TESTING

It is a non-functional test intended to test one of the software quality attributes i.e. “Scalability”. Scalability test is not focused on just one or few functionality of the software instead performance of software as a whole. Scalability testing is usually done by performance engineering team. Objective of scalability testing is to test the ability of the software to scale up with increased users, increased transactions, increase in database size etc., It is not necessary that software’s performance increases with increase in hardware configuration, scalability tests helps to find out how much more workload the software can support with expanding user base, transactions, data storage etc.,

STABILITY TESTING

It is a non-functional test intended to test one of the software quality attributes i.e. “Stability”. Stability testing focuses on testing how stable software is when it is subject to loads at acceptable levels, peak loads, loads generated in spikes, with more volumes of data to be processed. Scalability testing will involve performing different types of performance tests like load testing, stress testing, spike testing, soak testing, spike testing etc.…

Static Testing is a form of testing where in approaches like reviews, walkthroughs are employed to evaluate the correctness of the deliverable. In static

testing software code is not executed instead it is reviewed for syntax, commenting, naming convention, size of the functions and methods etc. Static testing usually has check lists against which deliverables are evaluated. Static testing can be applied for requirements, designs, and test cases by using approaches like reviews or walkthroughs.

Stress Testing is a type of performance testing, in which software is subjected to peak loads and even to a break point to observe how the software would behave at breakpoint. Stress testing also tests the behaviour of the software with insufficient resources like CPU, Memory, Network bandwidth, Disk space etc. Stress testing enables to check some of the quality attributes like robustness and reliability.

CHAPTER 12

12.SYSTEM IMPLEMENTATION

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended users and the operation of the system. The people are not sure that the software is meant to make their job easier.

- ✓ The active user must be aware of the benefits of using the system
- ✓ Their confidence in the software built up
- ✓ Proper guidance is impaired to the user so that he is comfortable in using the application

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not running on the server, the actual processes will not take place.

12.1 USER TRAINING

To achieve the objectives and benefits expected from the proposed system it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for education and training is more and more important. Education is complementary to training. It brings life to formal training by explaining the background to the resources for them. Education involves creating the right atmosphere and motivating user staff. Education information can make training more interesting and more understandable.

12.2 TRAINING ON THE APPLICATION SOFTWARE

After providing the necessary basic training on the computer awareness, the users will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design, type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the data entered. This training may be different across different user groups and across different levels of hierarchy.

OPERATIONAL DOCUMENTATION

Once the implementation plan is decided, it is essential that the user of the system is made familiar and comfortable with the environment. A documentation providing the whole operations of the system is being developed. Useful tips and guidance is given inside the application itself to the user. The system is developed user friendly so that the user can work the system from the tips given in the application itself.

SSYSTEM MAINTENANCE

The maintenance phase of the software cycle is the time in which software performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is to make adaptable to the changes in the system environment. There may be social, technical and other environmental changes, which affect a system which is being implemented. Software product enhancements may involve providing new functional capabilities, improving user displays and mode of interaction, upgrading the performance characteristics of the system. So only thru proper system maintenance procedures.

CORRECTIVE MAINTENANCE

The first maintenance activity occurs because it is unreasonable to assume that software testing will uncover all latent errors in a large software system. During the use of any large program, errors will occur and be reported to the developer. The process that includes the diagnosis and correction of one or more errors is called Corrective Maintenance.

ADAPTIVE MAINTENANCE

The second activity that contributes to a definition of maintenance occurs because of the rapid change that is encountered in every aspect of computing. Therefore Adaptive maintenance termed as an activity that modifies software to properly interfere with a changing environment is both necessary and commonplace

PERCEPTIVE MAINTENANCE

The third activity that may be applied to a definition of maintenance occurs when a software package is successful. As the software is used, recommendations for new capabilities, modifications to existing functions, and general enhancement are received from users. To satisfy requests in this category, Perceptive maintenance is performed. This activity accounts for the majority of all efforts expended on software maintenance.

PREVENTIVE MAINTENANCE

The fourth maintenance activity occurs when software is changed to improve future maintainability or reliability, or to provide a better basis for future enhancements. Often called preventive maintenance, this activity is characterized by reverse engineering and re-engineering techniques.

CHAPTER 13

13.CODING

```
import pandas as pd
import numpy as np
import seaborn as sns # data visualization library
import matplotlib.pyplot as plt
from sklearn.feature_selection import SelectKBest
from sklearn.preprocessing import MinMaxScaler
from sklearn.feature_selection import RFE
from sklearn.linear_model import LogisticRegression
#from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2
#from sklearn.preprocessing import MinMaxScaler
#from sklearn.svm import SVC
from sklearn.model_selection import train_test_split#,cross_val_score
from sklearn.metrics import
accuracy_score,confusion_matrix,classification_report
#from sklearn.ensemble import GradientBoostingClassifier
#from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
#from sklearn.linear_model import SGDClassifier
import warnings
warnings.filterwarnings('always')
from warnings import simplefilter
from sklearn.exceptions import DataConversionWarning
simplefilter(action='ignore', category=FutureWarning)
warnings.filterwarnings("ignore", category=DeprecationWarning)
warnings.filterwarnings(action='ignore', category=DataConversionWarning)
import warnings
warnings.simplefilter("ignore")
warnings.simplefilter("ignore", UserWarning)

df=pd.read_csv("phishing.csv")
print(df.head())
print(df.info())
df.isnull().sum()
X= df.drop(columns='class')
Y=df['class']
Y=pd.DataFrame(Y)
X.describe()

pd.value_counts(Y['class']).plot.bar()
plt.show()
#correlation map
f,ax = plt.subplots(figsize=(18, 18))
sns.heatmap(X.corr(), annot=True, linewidths=.5, fmt= '.1f',ax=ax)
```

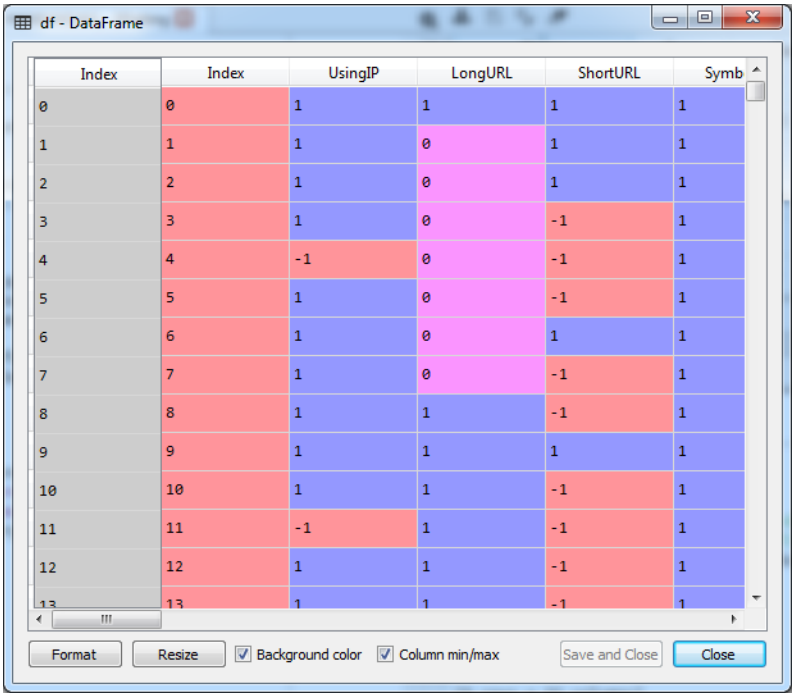
```
plt.show()

train_X,test_X,train_Y,test_Y =
train_test_split(X,Y,test_size=0.3,random_state=2)


X_norm = MinMaxScaler().fit_transform(X)

rfe_selector = RFE(estimator=LogisticRegression(), n_features_to_select=10,
step=10, verbose=5)
rfe_selector.fit(X_norm, Y)
rfe_support = rfe_selector.get_support()
rfe_feature = X.loc[:,rfe_support].columns.tolist()
print(str(len(rfe_feature)), 'selected features')
print(rfe_feature)

chi_selector = SelectKBest(chi2, k=10)
chi_selector.fit(X_norm, Y)
chi_support = chi_selector.get_support()
chi_feature = X.loc[:,chi_support].columns.tolist()
print(str(len(chi_feature)), 'selected features')
print(chi_feature)

# from sklearn.feature_selection import SelectFromModel
# from sklearn.linear_model import LogisticRegression
# embeded_lr_selector = SelectFromModel(LogisticRegression(penalty="l2"),
'1.25*median')
# embeded_lr_selector.fit(X_norm, Y)
# embeded_lr_support = embeded_lr_selector.get_support()
# embeded_lr_feature = X.loc[:,embeded_lr_support].columns.tolist()
# print(str(len(embeded_lr_feature)), 'selected features')
# print(embeded_lr_feature)

from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_norm = scaler.fit_transform(X)
from sklearn.decomposition import PCA
pca = PCA(n_components=25)
Y_sklearn = pca.fit_transform(X_norm)
cum_sum = pca.explained_variance_ratio_.cumsum()

pca.explained_variance_ratio_[:10].sum()

cum_sum = cum_sum*100

fig, ax = plt.subplots(figsize=(8,8))
plt.yticks(np.arange(0,110,10))
plt.bar(range(25), cum_sum, label='Cumulative _Sum_of_Explained _Varaince',
color = 'b',alpha=0.5)
```

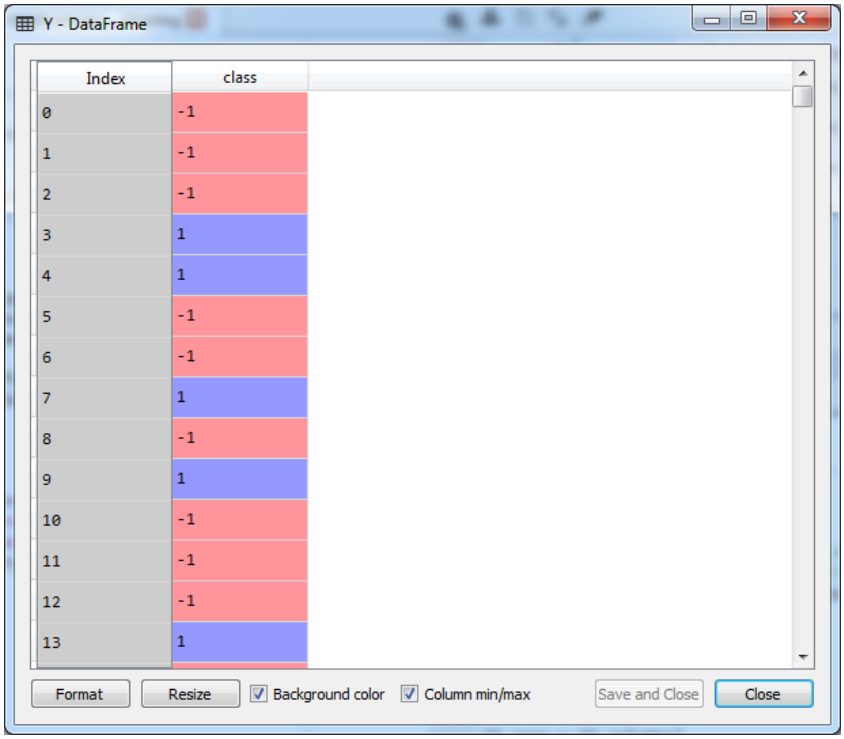
```
plt.title("Around 95% of variance is explained by the First 25 colmns ");
plt.show();


explained_variance=pca.explained_variance_ratio_
print(explained_variance.shape)
print(explained_variance.sum())
with plt.style.context('dark_background'):
    plt.figure(figsize=(6, 4))

    plt.bar(range(25), explained_variance, alpha=0.5, align='center',
            label='individual explained variance')
    plt.ylabel('Explained variance ratio')
    plt.xlabel('Principal components')
    plt.legend(loc='best')
    plt.tight_layout()
plt.show()

from sklearn.decomposition import TruncatedSVD
svd = TruncatedSVD(n_components=20, n_iter=50, random_state=42)
svd.fit(X_norm)
explained_variance=svd.explained_variance_ratio_
print(explained_variance.shape)
print(svd.explained_variance_ratio_.sum())
with plt.style.context('dark_background'):
    plt.figure(figsize=(6, 4))

    plt.bar(range(20), explained_variance, alpha=0.5, align='center',
            label='individual explained variance')
    plt.ylabel('Explained variance ratio')
    plt.xlabel('Principal components')
    plt.legend(loc='best')
    plt.tight_layout()
plt.show()

from sklearn.model_selection import learning_curve
def plot_learning_curve(estimator,title,X,y,ylim=None,cv=None,n_jobs=-
1,train_sizes=np.linspace(.1, 1.0, 5)):
    plt.figure()
    plt.title(title)
    if ylim is not None:
        plt.ylim(*ylim)
    plt.xlabel("Training Examples")
    plt.ylabel("Score")
    train_sizes, train_scores, test_scores = learning_curve(
        estimator, X, y, cv=cv, n_jobs=n_jobs, train_sizes=train_sizes)
    train_scores_mean = np.mean(train_scores, axis=1)
    train_scores_std = np.std(train_scores, axis=1)
    test_scores_mean = np.mean(test_scores, axis=1)
```

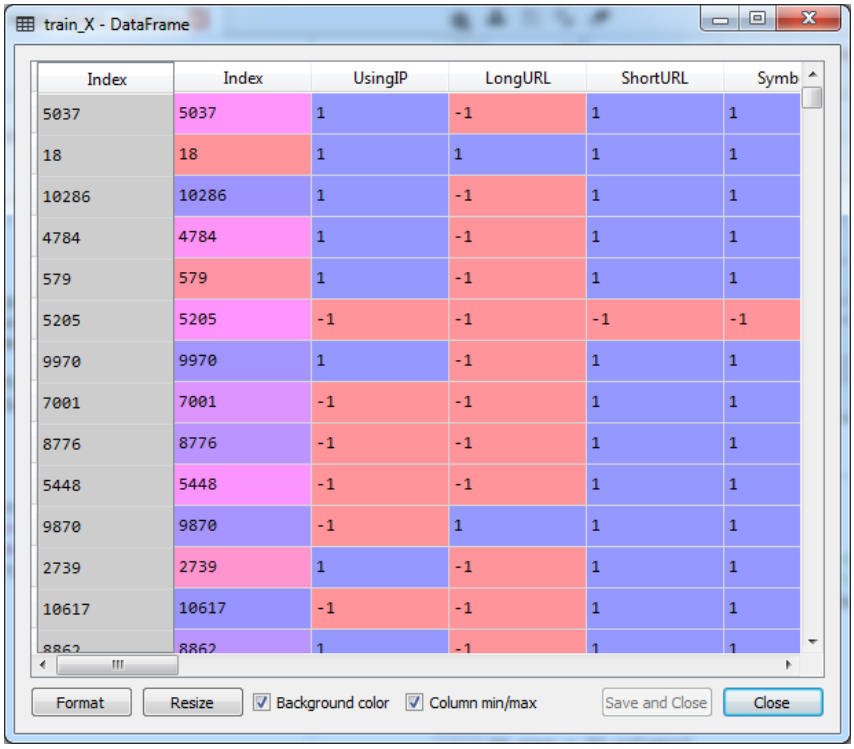
```
    test_scores_std = np.std(test_scores, axis=1)
    plt.grid()

    plt.fill_between(train_sizes, train_scores_mean - train_scores_std,
                     train_scores_mean + train_scores_std, alpha=0.1,
                     color="r")
    plt.fill_between(train_sizes, test_scores_mean - test_scores_std,
                     test_scores_mean + test_scores_std, alpha=0.1,
color="b")
    plt.plot(train_sizes, train_scores_mean, 'o-', color="r",
             label="Training score")
    plt.plot(train_sizes, test_scores_mean, 'o-', color="b",
             label="Cross-validation score")

    plt.legend(loc="best")
    return plt




''''RANDOM FOREST'''

print('RANDOM FOREST')
print('\n')
forest = RandomForestClassifier()
model6 = forest.fit(train_X,train_Y)
forest_predict = forest.predict(test_X)
acc_forest = accuracy_score(test_Y, forest_predict)
print('Random Forest Classifier Accuracy:',round(acc_forest*100,2))
print(classification_report(forest_predict,test_Y))
con  = confusion_matrix(forest_predict,test_Y)
plt.figure()
sns.heatmap(con,annot=True,fmt='.2f')
plt.show()
g = plot_learning_curve(model6," Random Forest Classifier learning
curves",train_X,train_Y
print(classification_report(knn_predict,test_Y))
con  = confusion_matrix(knn_predict,test_Y)
plt.figure()
sns.heatmap(con,annot=True,fmt='.2f')
plt.show()
g = plot_learning_curve(model7," KNN learning curves",train_X,train_Y)
plt.show()
```

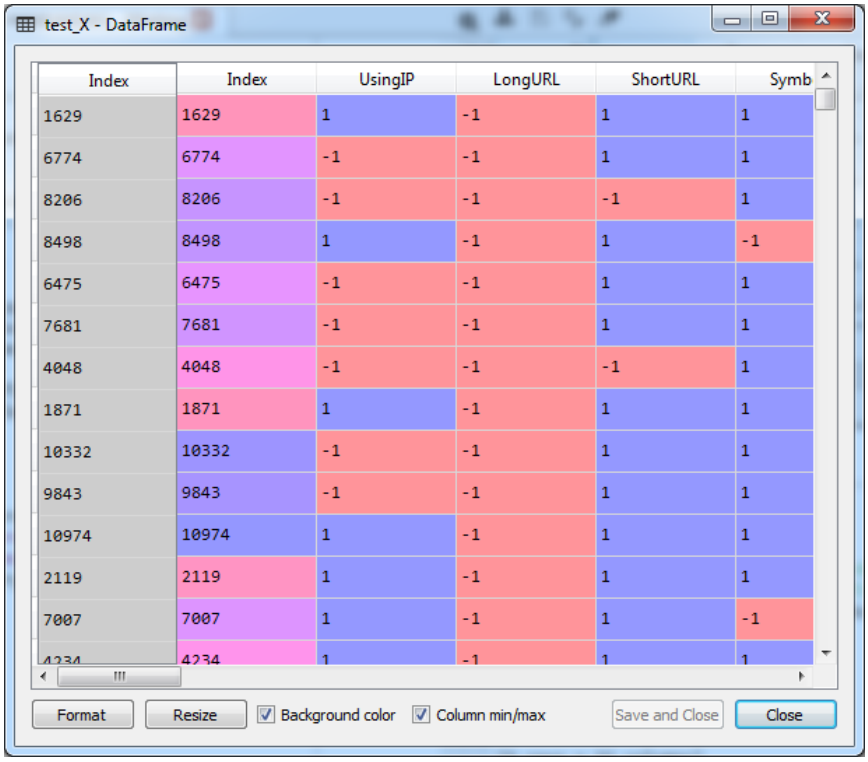
CHAPTER 14

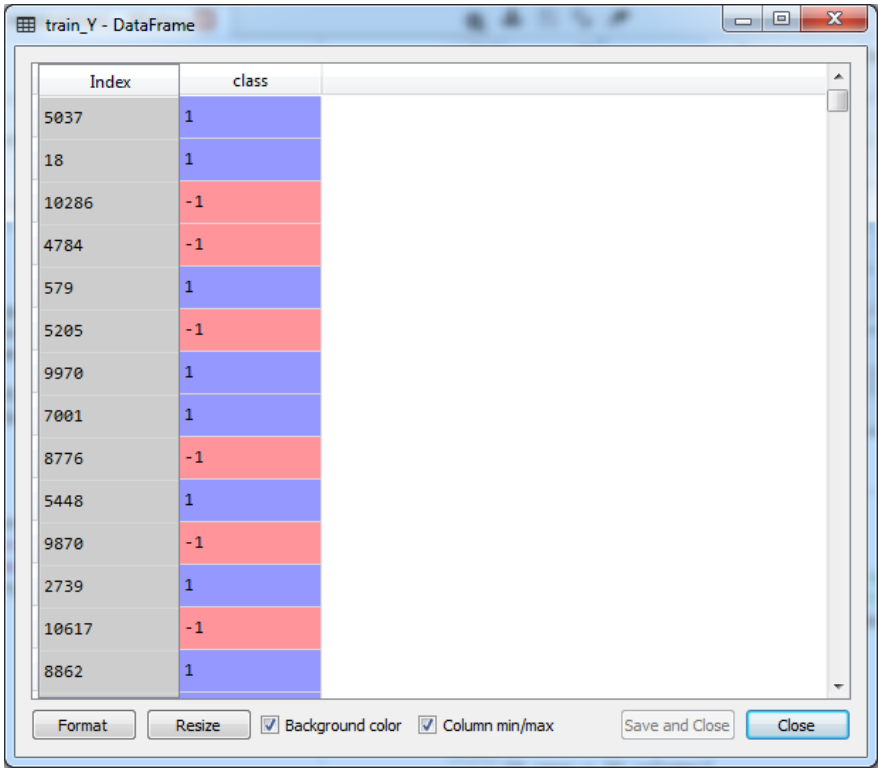
14.SCREEN SHOT&OUTPUT

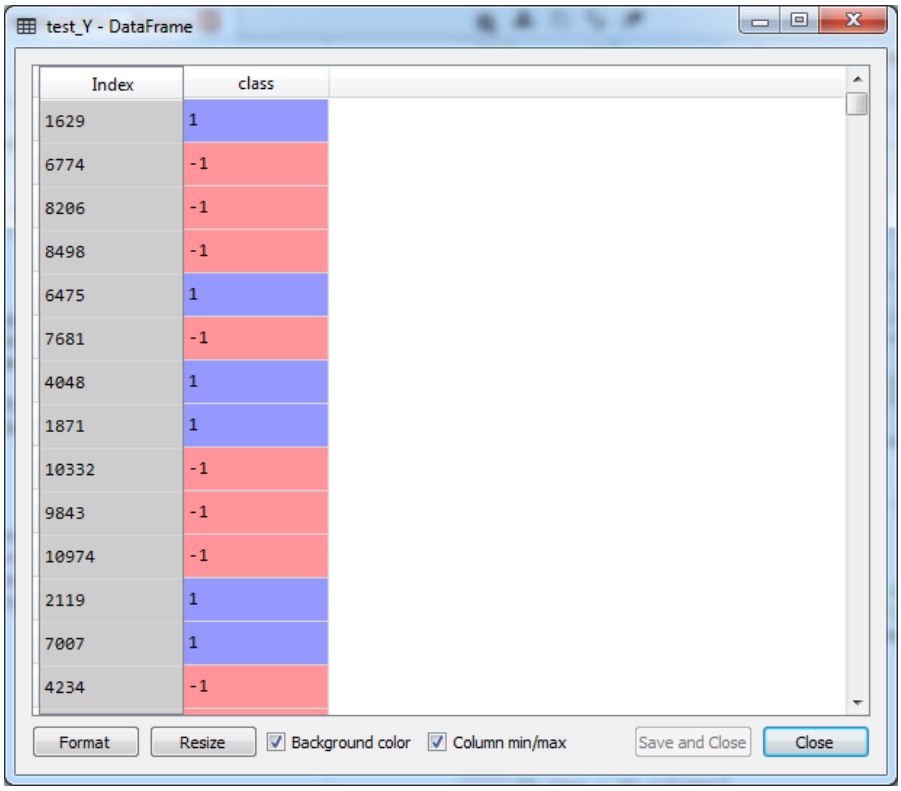


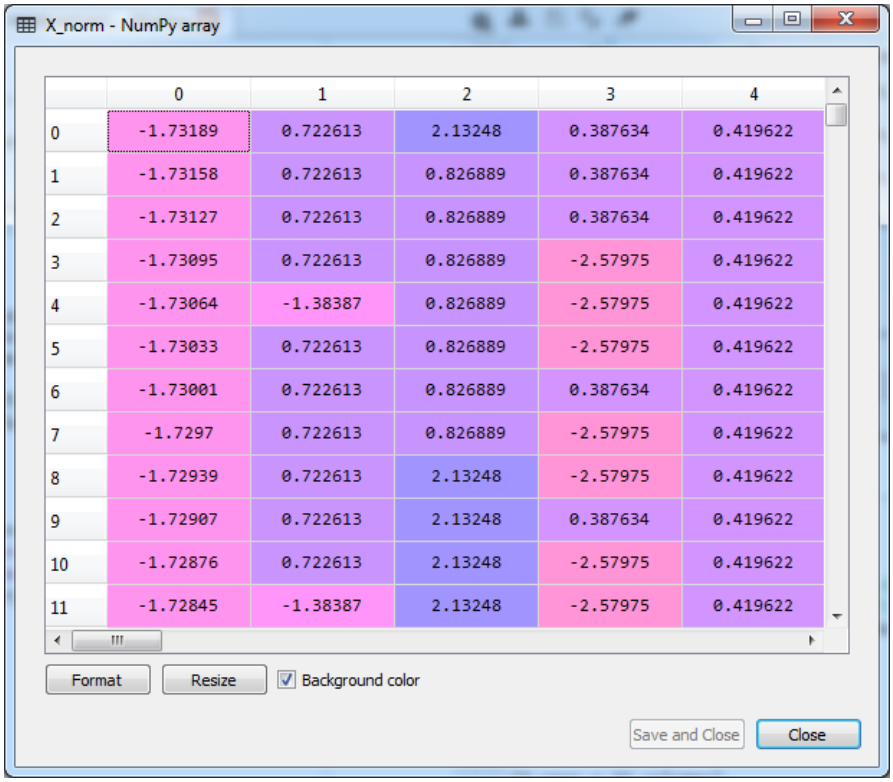



train_X - DataFrame
Index
Index
UsingIP
LongURL
ShortURL
Symb
5037 5037 1 -1 1 1
18 18 1 1 1 1
10286 10286 1 -1 1 1
4784 4784 1 -1 1 1
579 579 1 -1 1 1
5205 5205 -1 -1 -1 -1
9970 9970 1 -1 1 1
7001 7001 -1 -1 1 1
8776 8776 -1 -1 1 1
5448 5448 -1 -1 1 1
9870 9870 -1 1 1 1
2739 2739 1 -1 1 1
10617 10617 -1 -1 1 1
Format
Resize
Background color
Column min/max
Save and Close
Close

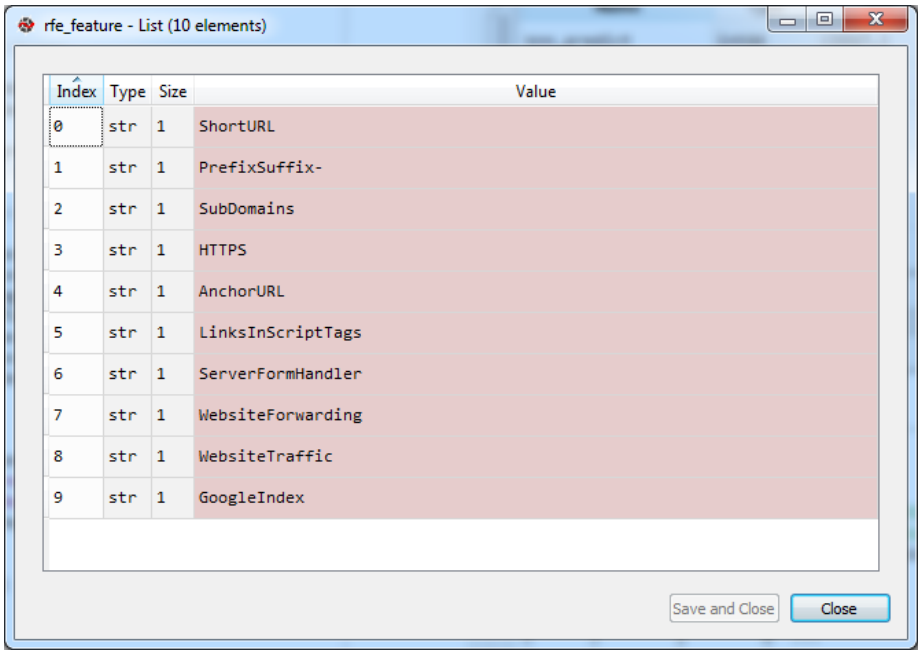

test_X - DataFrame
Index
Index
UsingIP
LongURL
ShortURL
Symb
1629 1629 1 -1 1 1
6774 6774 -1 -1 1 1
8206 8206 -1 -1 -1 1
8498 8498 1 -1 1 -1
6475 6475 -1 -1 1 1
7681 7681 -1 -1 1 1
4048 4048 -1 -1 -1 1
1871 1871 1 -1 1 1
10332 10332 -1 -1 1 1
9843 9843 -1 -1 1 1
10974 10974 1 -1 1 1
2119 2119 1 -1 1 1
7007 7007 1 -1 1 -1
Format
Resize
Background color
Column min/max
Save and Close
Close


train_Y - DataFrame
Index
class
5037 1
18 1
10286 -1
4784 -1
579 1
5205 -1
9970 1
7001 1
8776 -1
5448 1
9870 -1
2739 1
10617 -1
8862 1
Format
Resize
Background color
Column min/max
Save and Close
Close

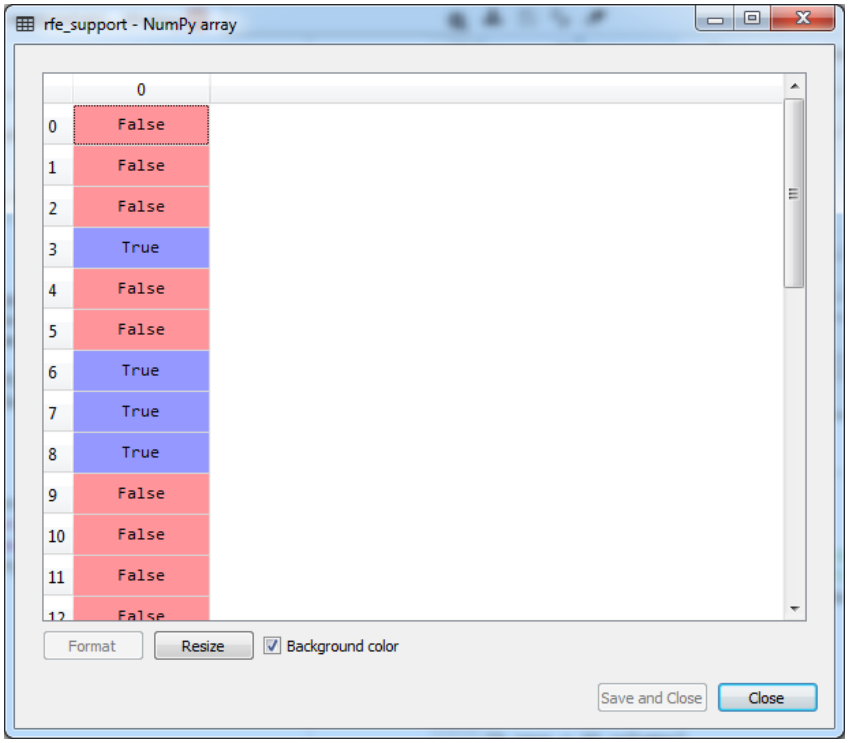

test_Y - DataFrame
Index
class
1629 1
6774 -1
8206 -1
8498 -1
6475 1
7681 -1
4048 1
1871 1
10332 -1
9843 -1
10974 -1
2119 1
7007 1
4234 -1
Format
Resize
Background color
Column min/max
Save and Close
Close

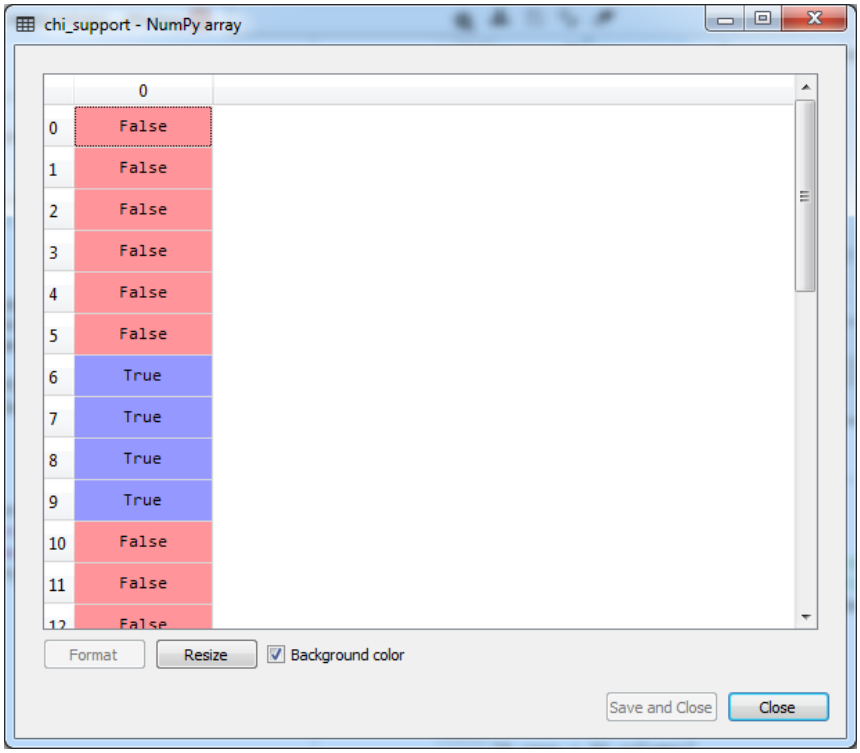


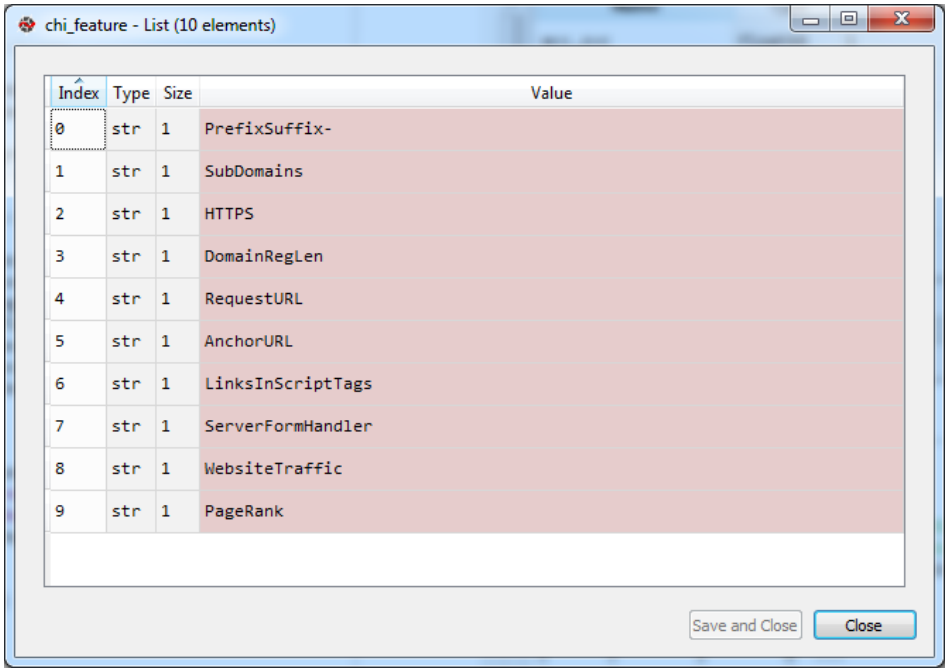
	0	1	2	3	4
0	-1.73189	0.722613	2.13248	0.387634	0.419622
1	-1.73158	0.722613	0.826889	0.387634	0.419622
2	-1.73127	0.722613	0.826889	0.387634	0.419622
3	-1.73095	0.722613	0.826889	-2.57975	0.419622
4	-1.73064	-1.38387	0.826889	-2.57975	0.419622
5	-1.73033	0.722613	0.826889	-2.57975	0.419622
6	-1.73001	0.722613	0.826889	0.387634	0.419622
7	-1.7297	0.722613	0.826889	-2.57975	0.419622
8	-1.72939	0.722613	2.13248	-2.57975	0.419622
9	-1.72907	0.722613	2.13248	0.387634	0.419622
10	-1.72876	0.722613	2.13248	-2.57975	0.419622
11	-1.72845	-1.38387	2.13248	-2.57975	0.419622

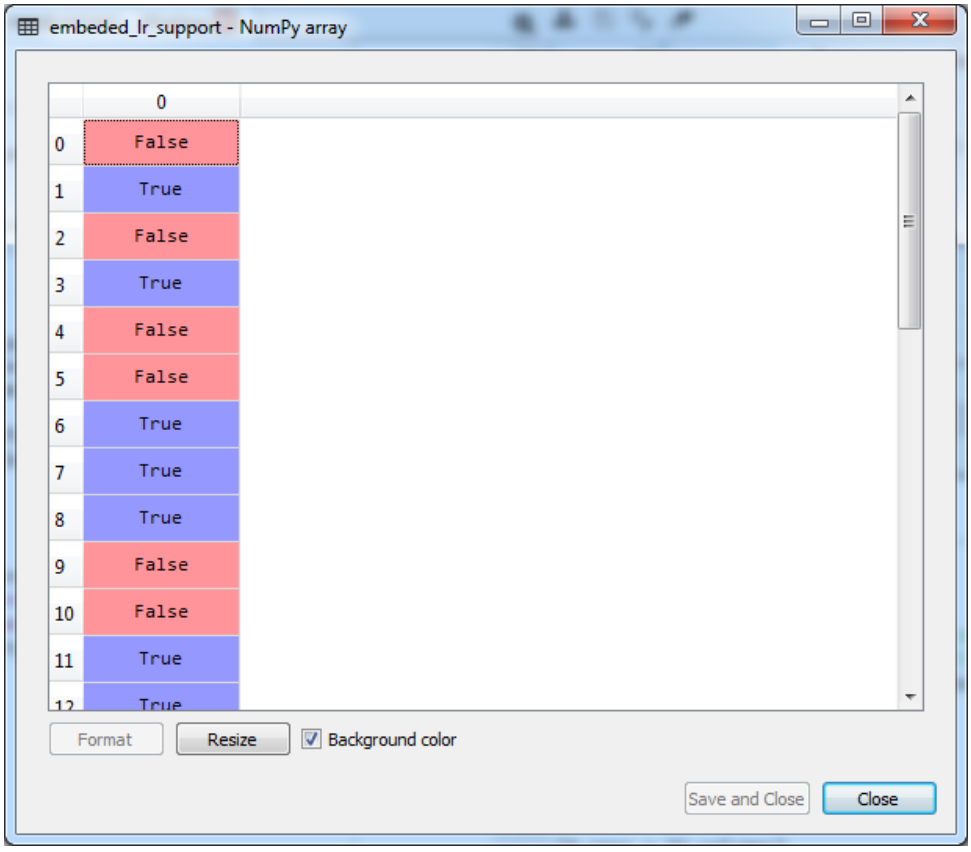


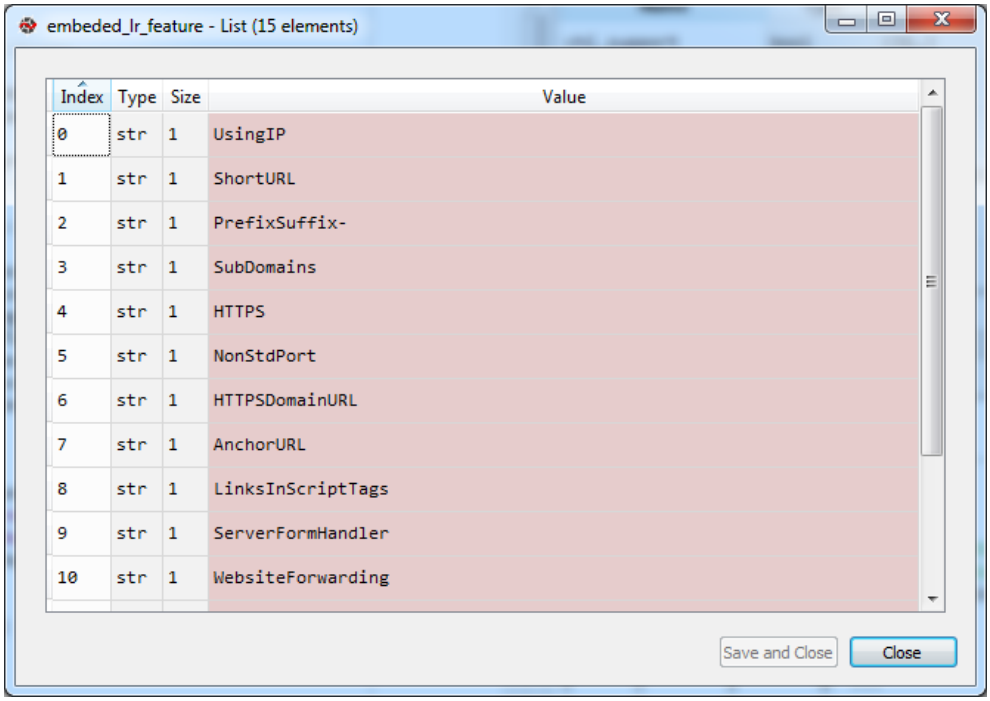
Index	Type	Size	Value
0	str	1	ShortURL
1	str	1	PrefixSuffix-
2	str	1	SubDomains
3	str	1	HTTPS
4	str	1	AnchorURL
5	str	1	LinksInScriptTags
6	str	1	ServerFormHandler
7	str	1	WebsiteForwarding
8	str	1	WebsiteTraffic
9	str	1	GoogleIndex

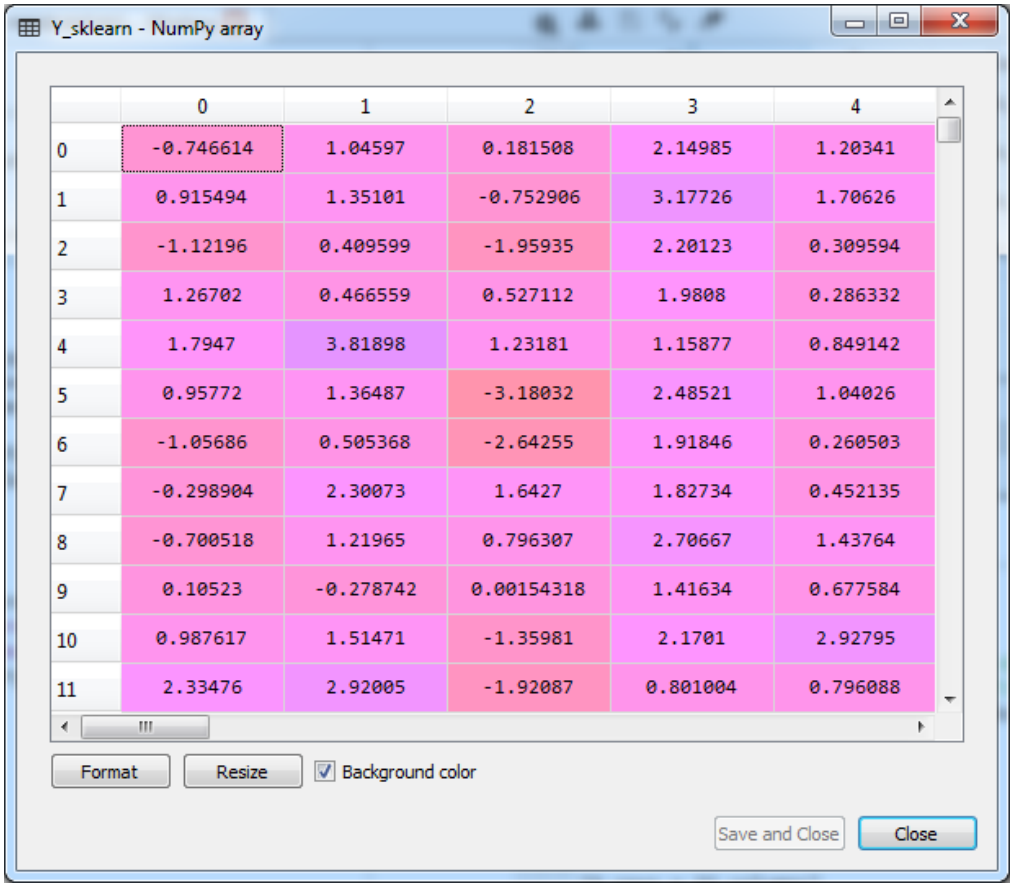

rfe_support - NumPy array
0
0 False
1 False
2 False
3 True
4 False
5 False
6 True
7 True
8 True
9 False
10 False
11 False
12 False
Format
Resize
Background color
Save and Close
Close

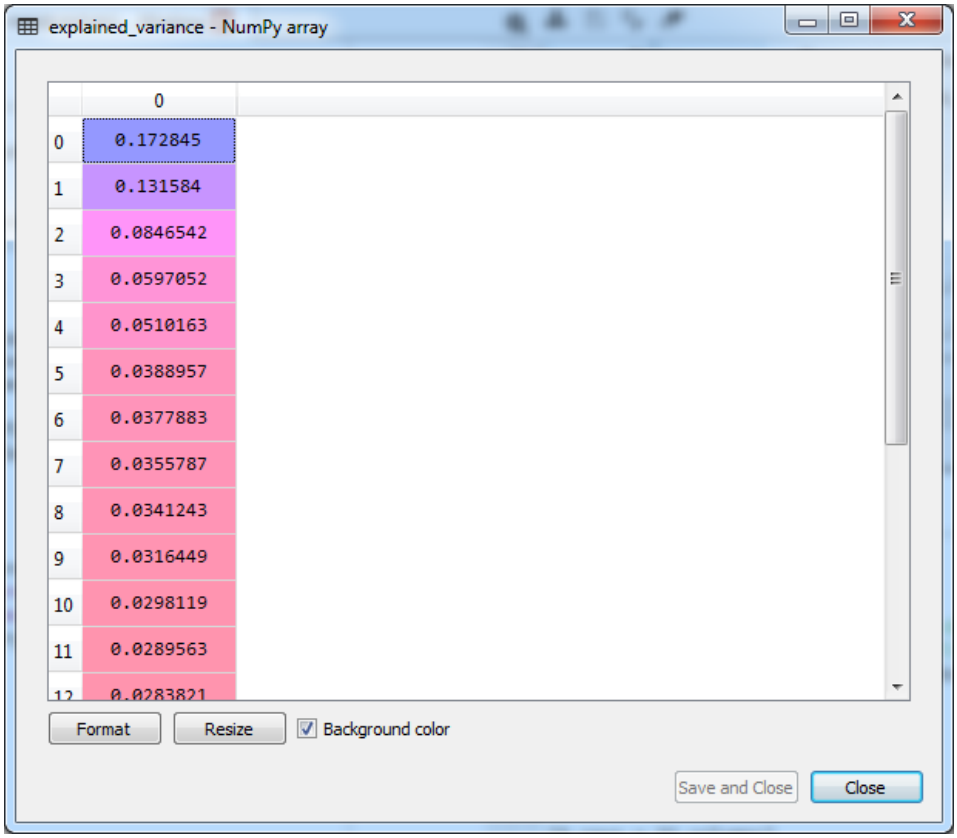

chi_support - NumPy array
0
0 False
1 False
2 False
3 False
4 False
5 False
6 True
7 True
8 True
9 True
10 False
11 False
12 False
Format
Resize
Background color
Save and Close
Close

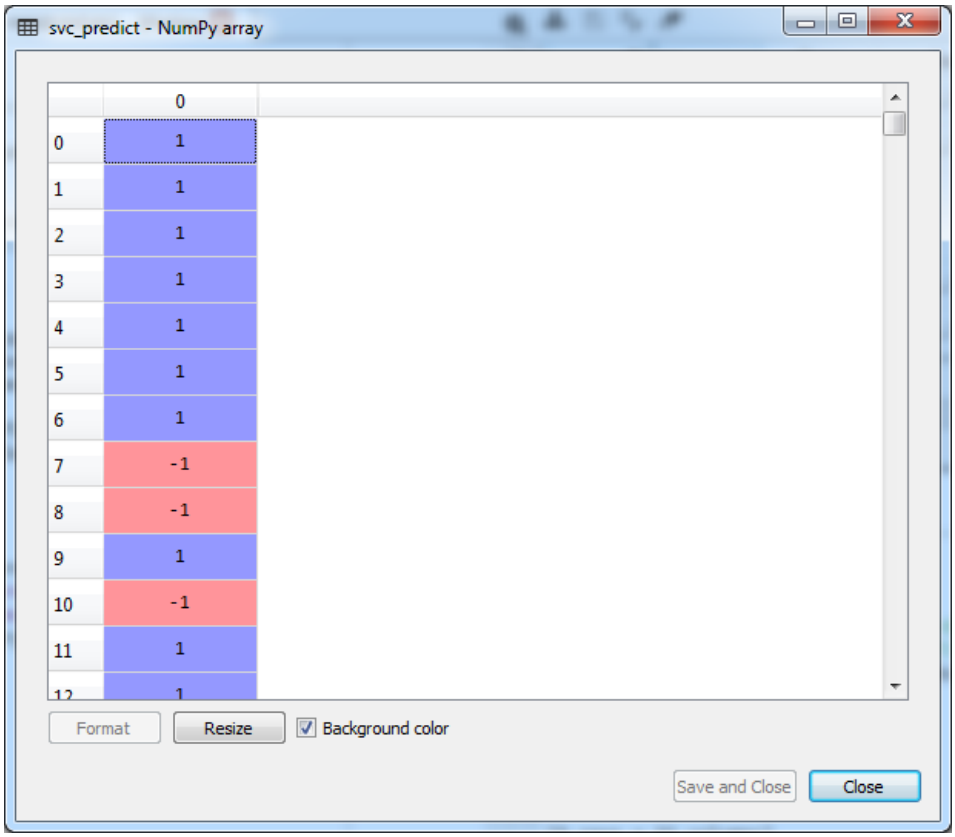

chi_feature - List (10 elements)
Index
Type
Size
Value
0 str 1 PrefixSuffix-
1 str 1 SubDomains
2 str 1 HTTPS
3 str 1 DomainRegLen
4 str 1 RequestURL
5 str 1 AnchorURL
6 str 1 LinksInScriptTags
7 str 1 ServerFormHandler
8 str 1 WebsiteTraffic
9 str 1 PageRank
Save and Close
Close

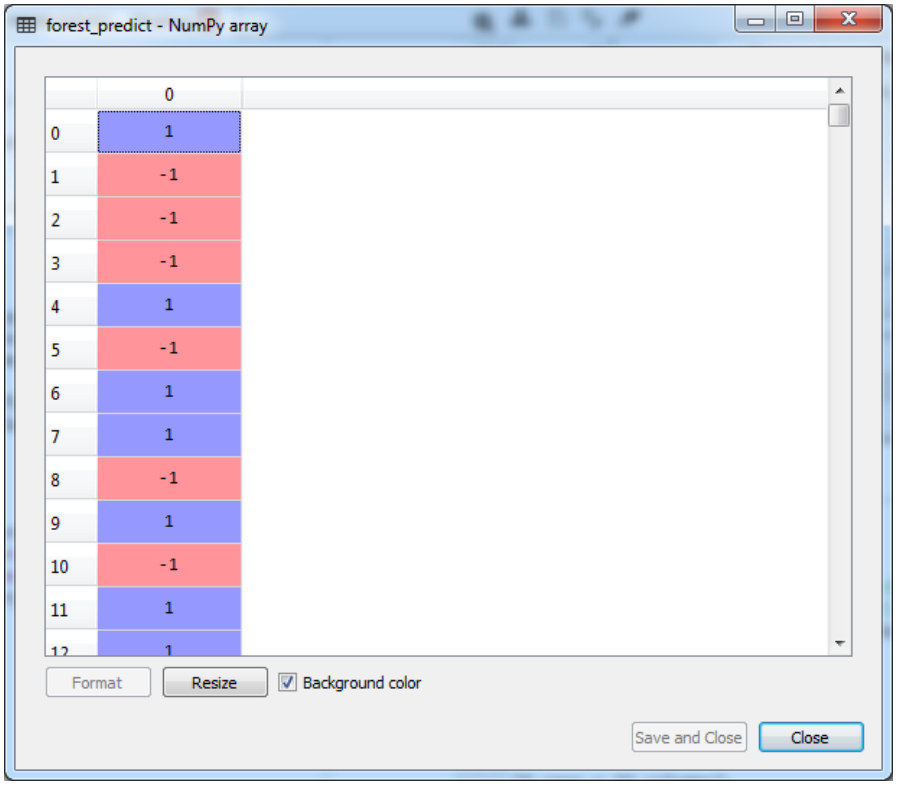

embeded_lr_support - NumPy array
0
0 False
1 True
2 False
3 True
4 False
5 False
6 True
7 True
8 True
9 False
10 False
11 True
Format
Resize
Background color
Save and Close
Close

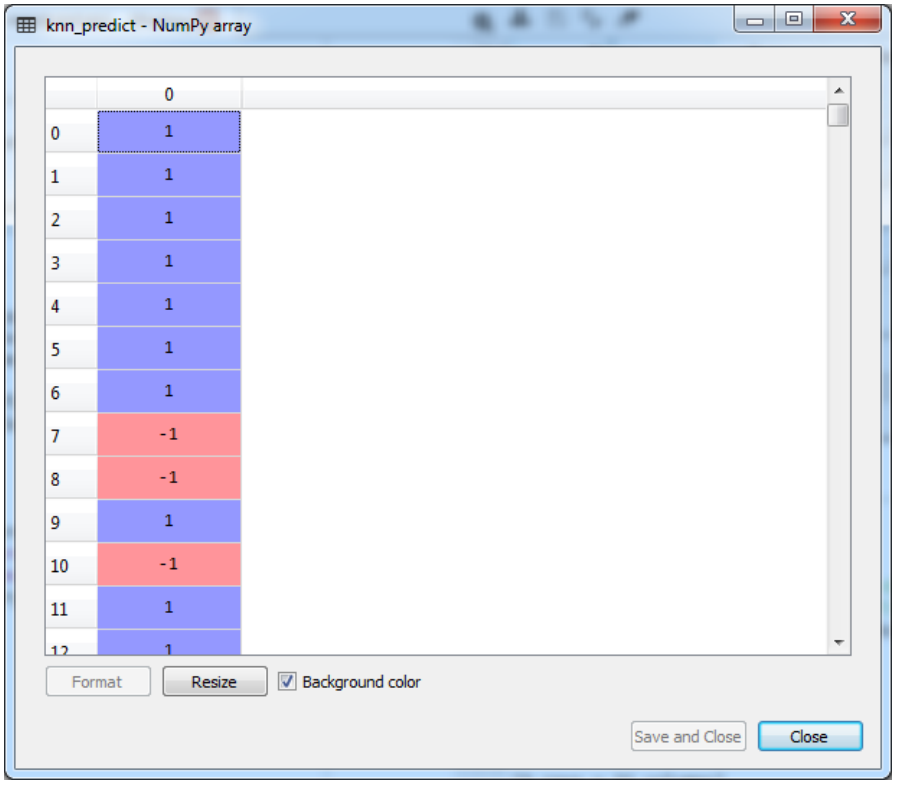

embeded_lr_feature - List (15 elements)
Index
Type
Size
Value
0 str 1 UsingIP
1 str 1 ShortURL
2 str 1 PrefixSuffix-
3 str 1 SubDomains
4 str 1 HTTPS
5 str 1 NonStdPort
6 str 1 HTTPSDomainURL
7 str 1 AnchorURL
8 str 1 LinksInScriptTags
9 str 1 ServerFormHandler
10 str 1 WebsiteForwarding
Save and Close
Close

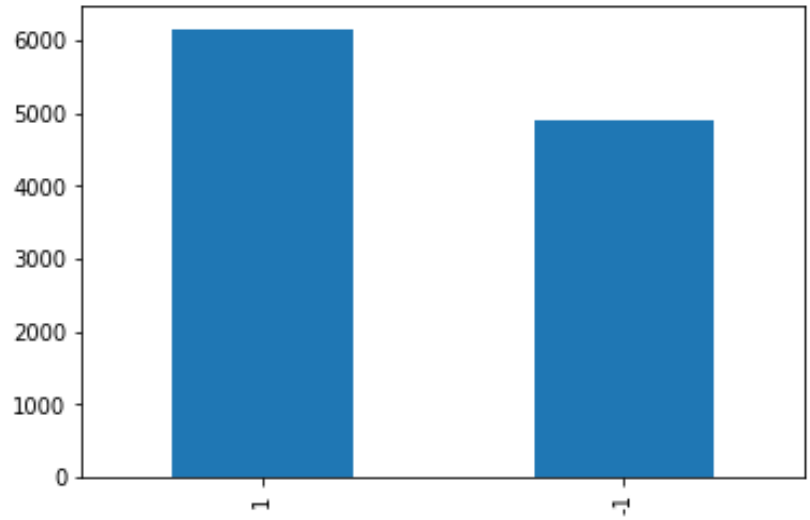

Y_sklearn - NumPy array
0 1 2 3 4
0 -0.746614 1.04597 0.181508 2.14985 1.20341
1 0.915494 1.35101 -0.752906 3.17726 1.70626
2 -1.12196 0.409599 -1.95935 2.20123 0.309594
3 1.26702 0.466559 0.527112 1.9808 0.286332
4 1.7947 3.81898 1.23181 1.15877 0.849142
5 0.95772 1.36487 -3.18032 2.48521 1.04026
6 -1.05686 0.505368 -2.64255 1.91846 0.260503
7 -0.298904 2.30073 1.6427 1.82734 0.452135
8 -0.700518 1.21965 0.796307 2.70667 1.43764
9 0.10523 -0.278742 0.00154318 1.41634 0.677584
10 0.987617 1.51471 -1.35981 2.1701 2.92795
11 2.33476 2.92005 -1.92087 0.801004 0.796088
Format
Resize
Background color
Save and Close
Close

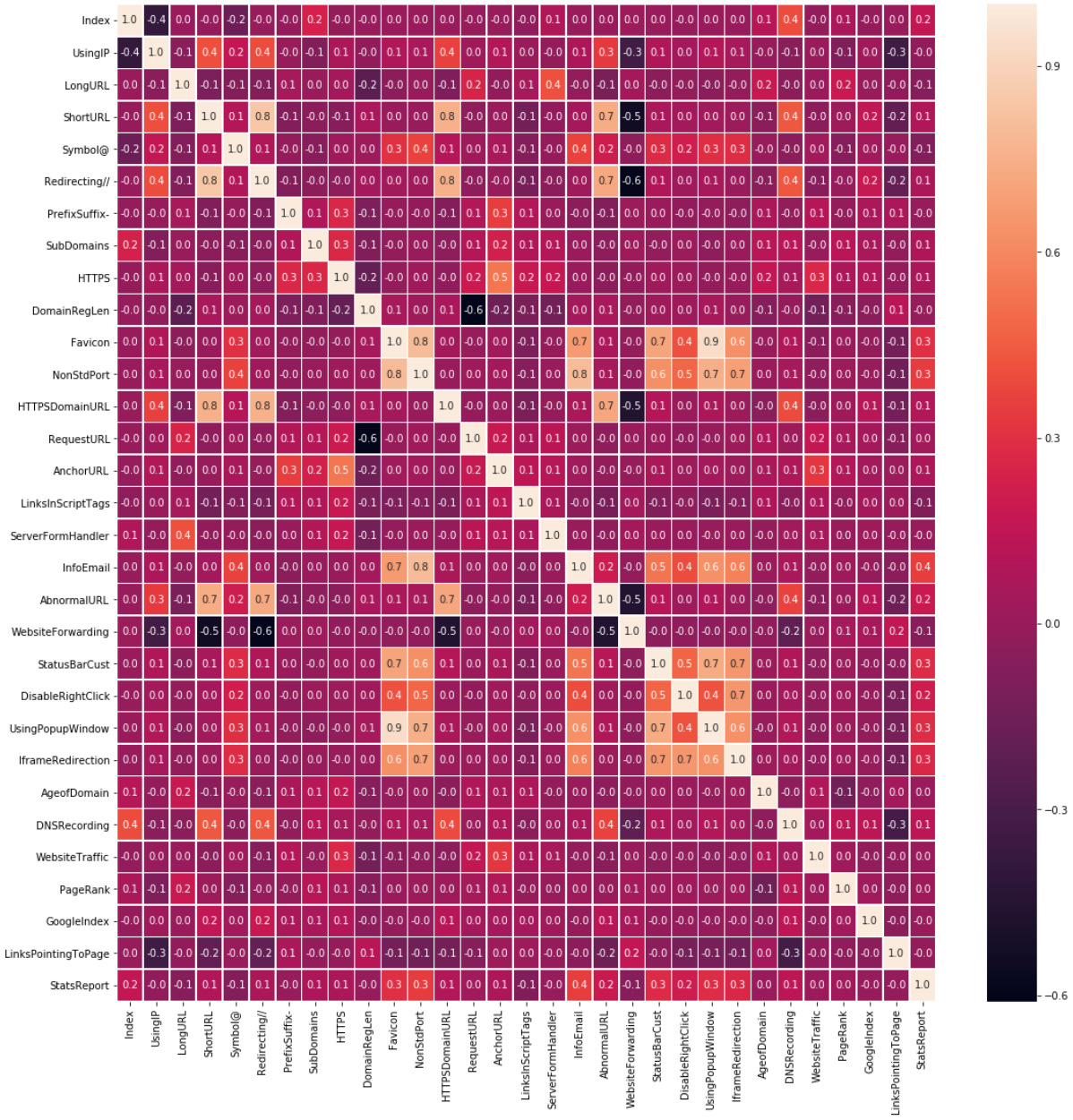

explained_variance - NumPy array
0
0 0.172845
1 0.131584
2 0.0846542
3 0.0597052
4 0.0510163
5 0.0388957
6 0.0377883
7 0.0355787
8 0.0341243
9 0.0316449
10 0.0298119
11 0.0289563
12 0.0283821
Format
Resize
Background color
Save and Close
Close


svc_predict - NumPy array
0
0 1
1 1
2 1
3 1
4 1
5 1
6 1
7 -1
8 -1
9 1
10 -1
11 1
12 1
Format
Resize
Background color
Save and Close
Close

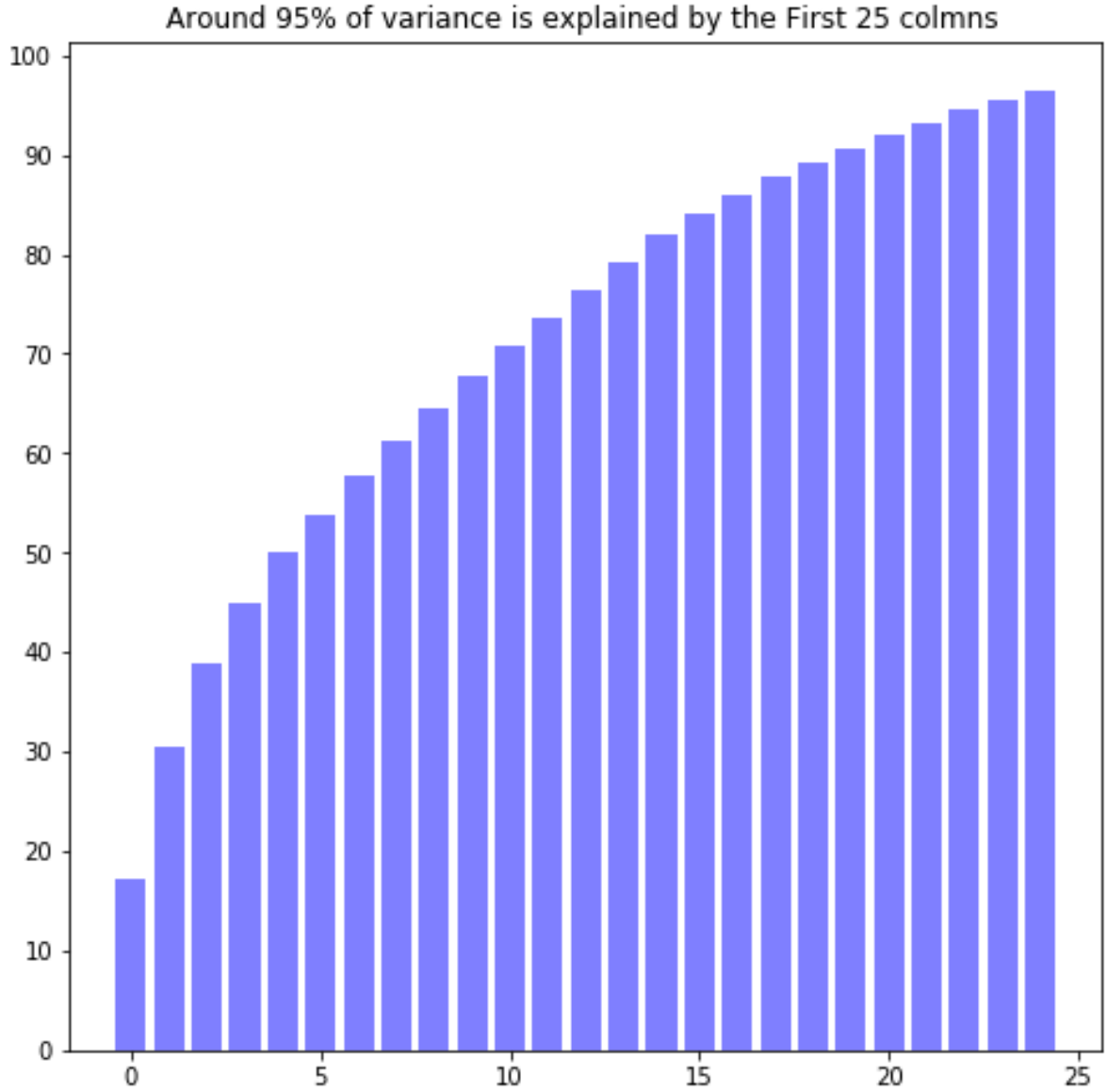

forest_predict - NumPy array
0
0 1
1 -1
2 -1
3 -1
4 1
5 -1
6 1
7 1
8 -1
9 1
10 -1
11 1
12 1
Format
Resize
Background color
Save and Close
Close

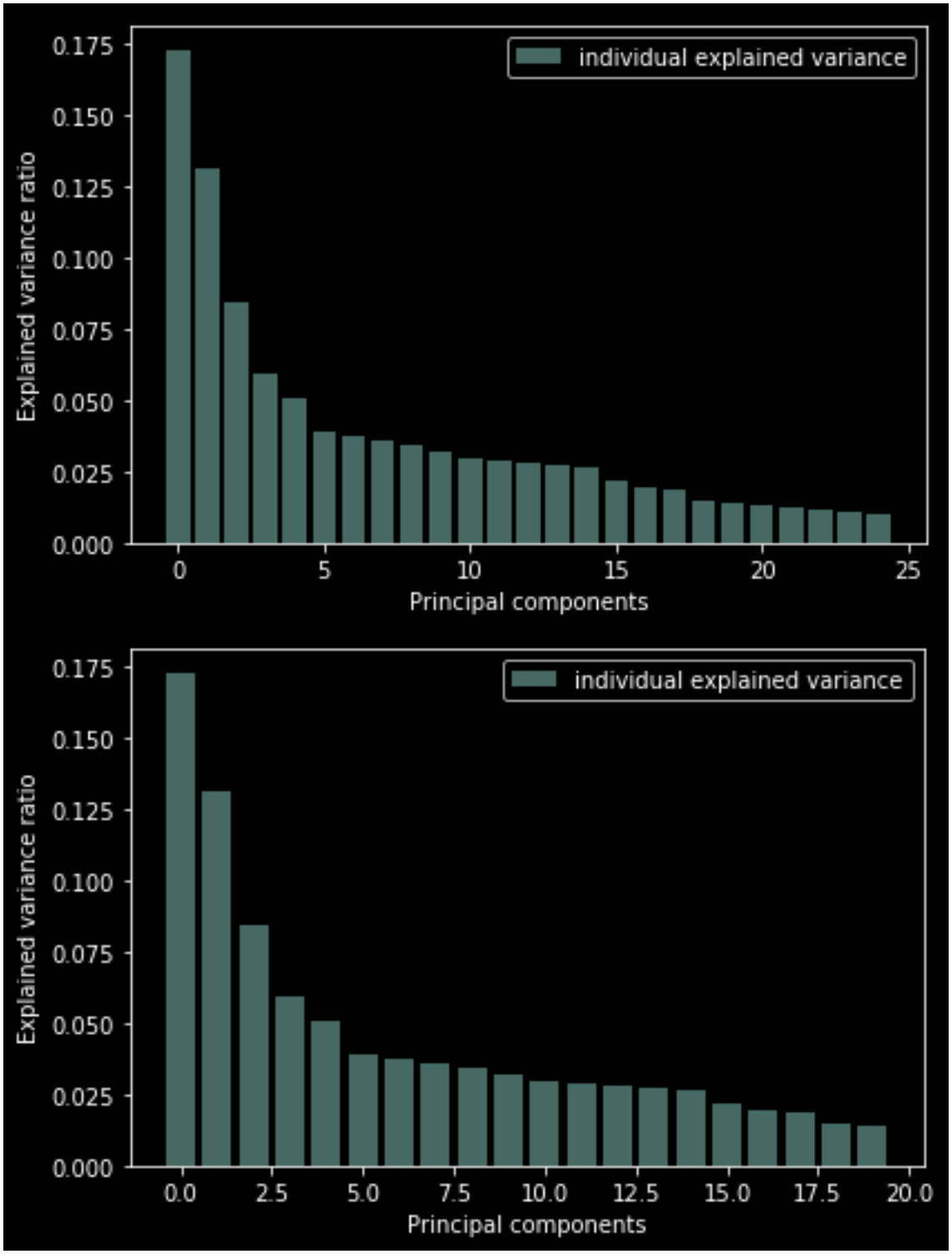

knn_predict - NumPy array
0
0 1
1 1
2 1
3 1
4 1
5 1
6 1
7 -1
8 -1
9 1
10 -1
11 1
12 1
Format
Resize
Background color
Save and Close
Close


6000
5000
4000
3000
2000
1000
0
1
-1

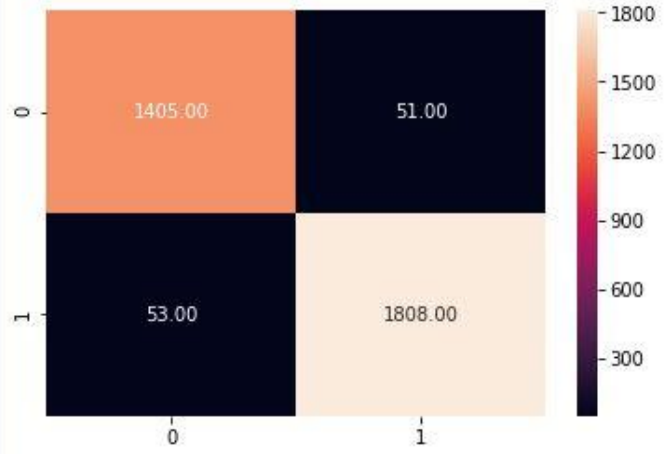

Index
UsingIP
LongURL
ShortURL
Symbol@
Redirecting//
PrefixSuffix-
SubDomains
HTTPS
DomainRegLen
Favicon
NonStdPort
HTTPSDomainURL
RequestURL
AnchorURL
LinksInScriptTags
ServerFormHandler
InfoEmail
AbnormalURL
WebsiteForwarding
StatusBarCust
DisableRightClick
UsingPopupWindow
IframeRedirection
AgeofDomain
DNSRecording
WebsiteTraffic
PageRank
GoogleIndex
LinksPointingToPage
StatsReport
0.9
0.6
0.3
0.0
−0.3
−0.6

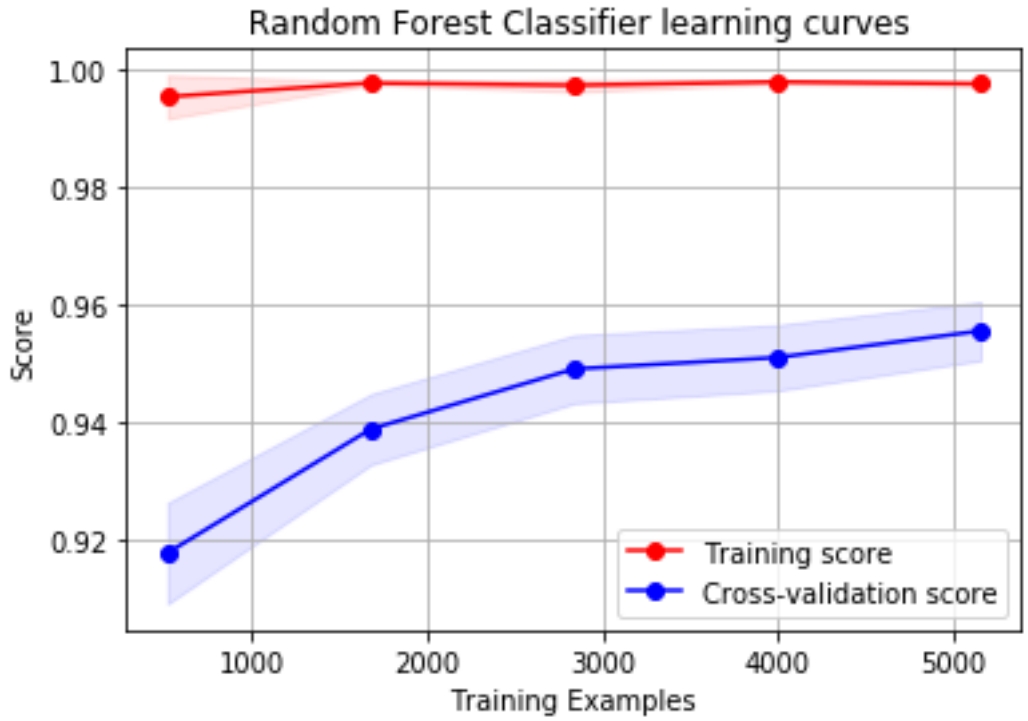
```
Fitting estimator with 31 features.
Fitting estimator with 21 features.
Fitting estimator with 11 features.
10 selected features
['ShortURL', 'PrefixSuffix-', 'SubDomains', 'HTTPS', 'AnchorURL', 'LinksInScriptTags',
'ServerFormHandler', 'WebsiteForwarding', 'WebsiteTraffic', 'GoogleIndex']
10 selected features
['PrefixSuffix-', 'SubDomains', 'HTTPS', 'DomainRegLen', 'RequestURL', 'AnchorURL',
'LinksInScriptTags', 'ServerFormHandler', 'WebsiteTraffic', 'PageRank']
15 selected features
['UsingIP', 'ShortURL', 'PrefixSuffix-', 'SubDomains', 'HTTPS', 'NonStdPort',
'HTTPSDomainURL', 'AnchorURL', 'LinksInScriptTags', 'ServerFormHandler',
'WebsiteForwarding', 'DNSRecording', 'WebsiteTraffic', 'GoogleIndex',
'LinksPointingToPage']
```



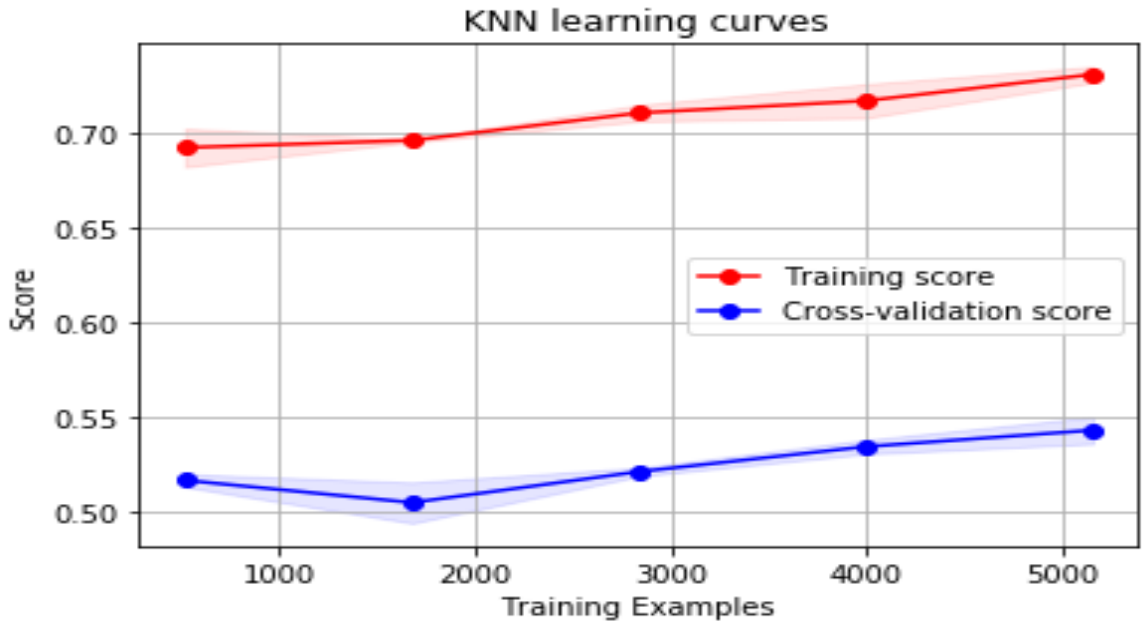

individual explained variance
Explained variance ratio
Principal components
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
0
5
10
15
20
25
individual explained variance
Explained variance ratio
Principal components
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

```
Random Forest Classifier Accuracy: 96.86
              precision    recall  f1-score   support

          -1       0.96      0.96      0.96      1456
           1       0.97      0.97      0.97      1861

   micro avg       0.97      0.97      0.97      3317
   macro avg       0.97      0.97      0.97      3317
weighted avg       0.97      0.97      0.97      3317
```





```
K-Nearest Neighbour Accuracy: 60.45
              precision    recall  f1-score   support

          -1       0.52      0.55      0.54      1380
           1       0.67      0.64      0.65      1937

   micro avg       0.60      0.60      0.60      3317
   macro avg       0.60      0.60      0.60      3317
weighted avg       0.61      0.60      0.61      3317
```



CHAPTER 15

15.FUTURE WORK

- In future, we will implement this process in different platforms.

- Data mining is one of the most widely used methods to extract data from different sources and organize them for better usage. In spite of having different commercial systems for data mining, a lot of challenges come up when they are actually implemented. With rapid evolution in the field of data mining, companies are expected to stay abreast with all the new developments.

CHAPTER 16

16.CONCLUSION

- This project Evaluate the performance of the classification and prediction.
- It finds the different classification performance and its summary effectively and quickly.
- It alleviate the sparsity problem and reduces the information loss.
- The accuracy of the classification result is highly increased.

REFERENCES

1. FBI, “Ic3 annual report released.”
2. APWG, “Phishing activity trends report.”
3. V. B. et al, “study on phishing attacks,” International Journal of Computer Applications, 2018.
4. I.-F. Lam, W.-C. Xiao, S.-C. Wang, and K.-T. Chen, “Counteracting phishing page polymorphism: An image layout analysis approach,” in International Conference on Information Security and Assurance, pp. 270–279, Springer, 2009.
5. W. Jing, “Covert redirect vulnerability,” 2017.
6. Krombholz .K., H. Hobel, M. Huber, and E. Weippl, “Advanced social engineering attacks,” Journal of Information Security and applications, vol. 22, pp. 113–122, 2015.
7. P. Kumaraguru, J. Cranshaw, A. Acquisti, L. Cranor, J. Hong, M. A. Blair, and T. Pham, “School of phish: a real-world evaluation of anti-phishing training,” in Proceedings of the 5th Symposium on Usable Privacy and Security, pp. 1–12, 2009.
8. R. C. Dodge Jr, C. Carver, and A. J. Ferguson, “Phishing for user security awareness,” computers & security, vol. 26, no. 1, pp. 73–80, 2007.

9. R. Dhamija, J. D. Tygar, and M. Hearst, “Why phishing works,” in Proceedings of the SIGCHI conference on Human Factors in computing systems, pp. 581–590, 2006.
10. C. Ludl, S. McAllister, E. Kirda, and C. Kruegel, “On the effectiveness of techniques to detect phishing sites,” in International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, pp. 20–39, Springer, 2007.
11. P. Rosiello, E. Kirda, F. Ferrandi, et al., “A layout-similarity-based approach for detecting phishing pages,” in 2007 Third International Conference on Security and Privacy in Communications Networks and the Workshops-SecureComm 2007, pp. 454–463, IEEE, 2007.
12. S. Afroz and R. Greenstadt, “Phishzoo: Detecting phishing websites by looking at them,” in 2011 IEEE fifth international conference on semantic computing, pp. 368–375, IEEE, 2011.
13. K.-T. Chen, J.-Y. Chen, C.-R. Huang, and C.-S. Chen, “Fighting phishing with discriminative keypoint features,” IEEE Internet Computing, vol. 13, no. 3, pp. 56–63, 2009.
14. K. Jain and B. B. Gupta, “Phishing detection: Analysis of visual similarity based approaches,” Security and Communication Networks, vol. 2017, 2017.

15.sR. S. Rao and S. T. Ali, “A computer vision technique to detect phishing attacks,” in 2015 Fifth International Conference on Communication Systems and Network Technologies, pp. 596–601, IEEE, 2015.

16.B. Gupta, N. A. Arachchilage, and K. E. Psannis, “Defending against phishing attacks: taxonomy of methods, current issues and future directions,” Telecommunication Systems, vol. 67, no. 2, pp. 247–267, 2018.

17.Karatzoglou, D. Meyer, and K. Hornik, “Support vector machines in r,” Journal of statistical software, vol. 15, no. 9, pp. 1–28, 2006.

18.Breiman, “Random forests,” Machine learning, vol. 45, no. 1, pp. 5– 32, 2001.

19.T. Hastie, S. Rosset, J. Zhu, and H. Zou, “Multi-class adaboost,” Statistics and its Interface, vol. 2, no. 3, pp. 349–360, 2009.

20.J. H. Friedman, “Stochastic gradient boosting,” Computational statistics & data analysis, vol. 38, no. 4, pp. 367–378, 2002.