TIEVS – FRAUDULENT CONTENT DETECTION USING MACHINE LEARNING APPLIANCES

Ravihari J.M.S

(IT18089400)

BSc (Hons) in Information Technology Specializing in Information Technology

Department of Information Technology

Sri Lanka Institute of Information Technology
Sri Lanka

October 2021

TIEVS – FRAUDULENT CONTENT DETECTION USING MACHINE LEARNING APPLIANCES

Ravihari J.M.S

(IT18089400)

Dissertation Submitted in Partial Fulfillment of the Requirements for the BSc (Hons) in Information Technology Specializing in Information Technology

Department of Information Technology

Sri Lanka Institute of Information Technology
Sri Lanka

October 2021

DECLARATION

We declare that this is our own work and this dissertation1 does not incorporate without acknowledgment any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgment is made in the text.

Also, I hereby grant to Sri Lanka Institute of Information Technology, the non-exclusive right to reproduce and distribute our dissertation, in whole or in part in print, electronic, or another medium. We retain the right to use this content in whole or part in future works (such as articles or books).

Name	Student ID	Signature
Ravihari J.M.S	IT18089400	Ravihari

The above candidates have carried out research for the bachelor's degree Dissertation under my supervision.

Signature of the supervisor: Date:

ABSTRACT

Due to the obvious scarcity of physical magazines, online classified advertising has grown in popularity. Nonetheless, current platforms are cumbersome. The majority of systems are created using traditional technologies that are less scalable, less accurate, and have high latency processes. Furthermore, buyers find it exhausting when it comes to defining trustworthy, authentic classified advertising selling postings. Furthermore, strict validation procedures for detecting and preventing fraudulent material from being published in the systems have been neglected. Consequently, this is extensively focused on creating an intelligent fraud content detection system, especially targeting the products for sale on our 'Tievs' online classified advertising platform, especially Cars. Due to comparative cost and the widespread adoption of the renting culture, the population of used cars has been rising at an increasing rate. Having fake description analysis with excellent accuracy in proactively identify ads legitimacy to cater to hassles faced by customers, was satisfied. This study was initiated to evaluate the expertise of 7 ML algorithms such as Decision Tree, RF, SVM, KNN, XGBoost, XGBRF, and Light Gradient Boosting Machine (LightGBM) classifier in a comparative analysis after vectorization. Optimal Light Gradient Boosting classifier-driven was proven to have 95.79% of precision after pertinent parameters out of analyzed models through supervised learning to identify fraud advertisement context prior to ad submission. Perhaps some tweaking to the training phase with a larger fraud content dataset, presuming to augment recall rate precisely while reducing precision loss, could be done in the foreseeable future. Hence, the proposed solution’s objective of surpassing former classified advertising fraud detection systems in delivering customers’ satisfactory and genuine selling feed, using the timesaving, human-centric, and error-preventive approaches, was accomplished.

Key Words – Fraud detection, LightGBM, Supervised Marching Learning , Used Car Content Detection , Classfied Advertisi

ACKNOWLEDGEMENT

I would like to highly appreciate the continued guidance and expressing my heartfelt gratitude to our supervisor, Ms. Manori Gamage (Lecturer -Faculty of Computing SLIIT), and co-supervisor, Ms. Suriyaa Kumari (LECTURE -Faculty of Computing SLIIT), for thier suggestions and assistance in composing this paper. And I would like to express my deepest appreciation to her for their unwavering support, encouragement, and enthusiasm during the research, as well as their insightful comments on how to make the 'TIEVS' system more efficient and productive.

Moreover, Ms. Wasanthi Bandaranayake is implicitly thanked for expressing her positive and negative experiences utilizing online classified advertising platforms, as well as speculative modifications which should have been integrated inside an advanced technological aspect. Her thoughts and suggestions paved the way for a new study route that has yet to be explored and researched in the current literature in the disciplines of information technology and business.

And also grateful to Sri Lanka Institute of Information Technology lecturers, research team members and family members for their ongoing guidance and encouragement and also Mr. Austin Reese, Mr. Baris Dincer, and the Kaggle team deserve special recognition for generating and making analysis-foundational datasets freely available to all scholars and tech enthusiasts.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Description
ML	Machine Learning
API	Application Programming Interface
LightGBM	Light Gradient Boosting Machine
UI	User Interface
RF	Random Forest
XGBoost	Extreme Gradient Boosting
DT	Decision Tree
XGBRF	Extreme Gradient Boosting Random Forest
KNN	k-Nearest Neighbours
SVM	Support Vector Machine

1 INTRODUCTION

1.1 Background

At present, the vast majority of the population does not consume tangible sources such as newspapers, magazines, booklets, leaflets, and so forth. Especially now since the globe is in the grip of a pandemic, it is much more convenient and safer, even without the necessity to travel long distances and exchange money with other individuals, simply selling or buying through advertisement post(s), without exposing our health and safety at risk. Since the moment, online classified advertising such as Craiglist, CarDheko, ikman.lk, Quikr and Olx have drawn a significant number of buyers and sellers since online classified advertisements have enhanced an essential element of the advertisement market as they are convenient and straightforward to operate, and they could target a significant proportion of individuals in a user's surrounding neighborhood, for instance, Craigslist is a significant platform in the United States; on this sector, consumers may publish a few photographs, include a brief description, and end up selling them. They can freely advertise their advertisements on the internet. According to market researcher classified intelligence, the U.S market for online classified advertisement was $14.1 billion in 2003, and it has increased quickly since then [1].

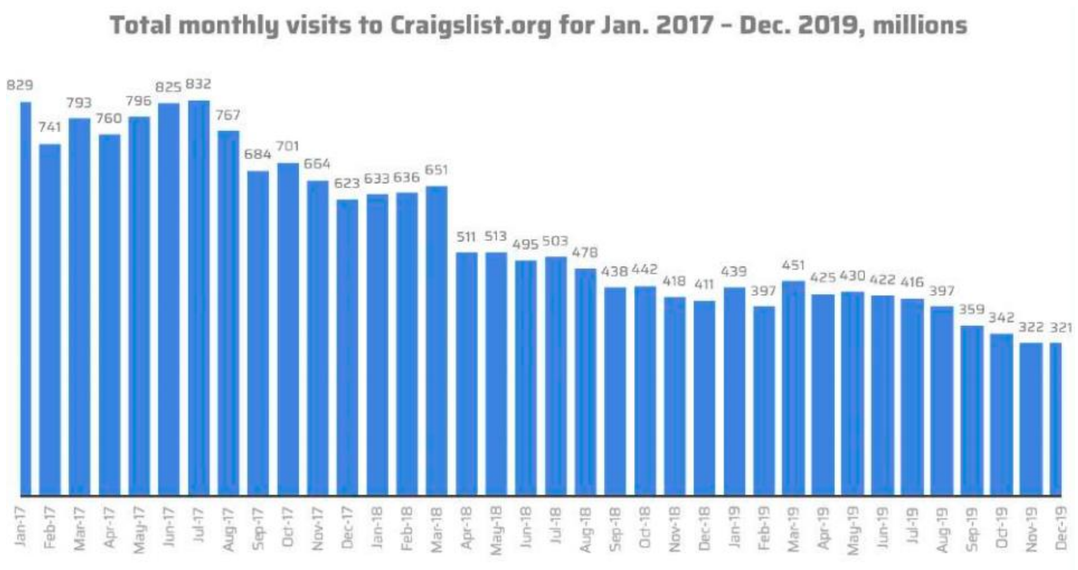


Figure 1.1 Monthly visits to cragslist.org

Craigslist, for instance, receives millions of new posts every month as depicted in Fig 1-1, and ranked the 7th most visited site in the U.S And the 35th most visited site in the world, according to alexa2[1].

As depicted in Fig 1-2, the use of online classified ads sites doubled, according to the pew internet and American life project survey which was conducted based on the internet users on 26th of March -29th of April in 2009, ensuing that economical online classified advertisement sites such as aforementioned classifieds sites have attracted plenty of posts and visits. Additionally, comparative affordability and the widespread adoption of the rental culture, the population of used cars has been rising at an increasing rate. Currently, classified ads can reach more target groups using online classified advertising platforms, through the advent of sophisticated technology. Nonetheless, certain circumstances may arise in which buyers are not entirely confident in purchasing a product and, as a result, may presumably undertake comprehensive analysis on how to purchase and confirm a trustworthy product. The growth of the Internet, ease of communication, and recent technology covered the way criminals conduct fraudulent actions which consequence of the loss of dollars globally each year.

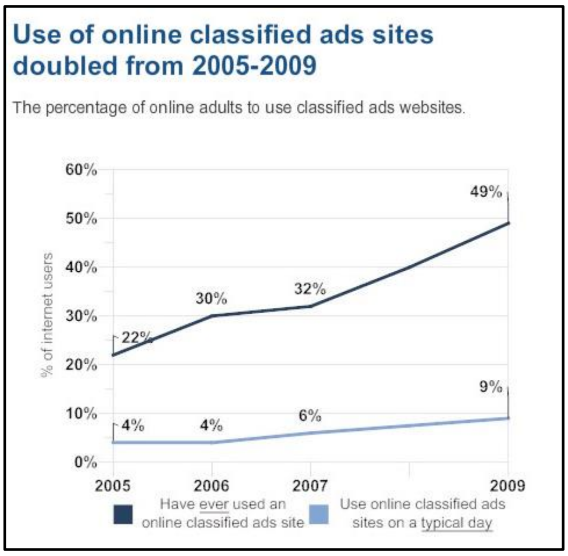


Figure 1.2 Use of online classified ads

Due to its high commercial potential, the online classified advertisement domain is a target for spammers, and this has become one of the biggest issues hindering further development of online advertisement [1]. The populace of the used or second-hand vehicles on the global market has been booming ever since, and it is conceivable that it has more than doubled over the last few years, due to the Covid-19 epidemic. However, as with many other pandemic surprising accomplishments, the moment of uncertainty and personal distress has given rise to a boom in the sale and purchase of old cars. With a paucity of new automobiles from automobile manufacturers able to access older cars by April and May, and consumers more hesitant about investing in large things, used vehicle sales soared. This trend has persisted throughout the summer and fall: according to Edmunds Edmunds.com data., the last two months, August and September, had the fastest growth rate of used car sales volume in the previous 18 months.

Besides, Since sites attract a massive amount of advertisements on daily basis, each person who desires to use such a platform can create an account and submit something they wish. However, due to the fact, that platforms are each unfastened and very smooth to use, scammers additionally utilize them to post fake advertisements, consequently, detection of fraudulent classified advertisement content has been an interesting topic among researchers. Fraudsters have taken advantage of the rising number of consumers on those sites. According to a State University Of New York research of Craigslist residential advertising in San Francisco, 1.5 percentage of the housing advertisements are considered fraudulent or misleading [2]. According to another survey, 43 percent of renters have been perpetrators of, or have been exposed to, rented scams in residential ads [3]. Annually, such frauds cost digital ad networks and buyers millions of dollars [4]. Consumers who utilize classified ad platforms to buy and/or sell items could efficaciously preserve themselves by comprehending how fraudsters perceive and operate. Scammers employ a variety of malevolent clues to deceive the public; nevertheless, most of the time, there are clues of the fraudulent source that may assist in determining the trustworthiness of an advertisement. Typically, fraudsters do not even have the listed products to sell or, in other circumstances, have the listed item but it is in inadequate form. Consequently, fraudsters frequently offer superior photographs of the products they are advertising is intended to mislead potential consumers. Considering buyers might be misled by unrealistic advertisements and falling into misconceptions, it is indisputable that an efficient and appropriate utilized fraud detection system is an inescapable necessity to establish genuine appropriateness for an advertisement, and with no dispensable obstacles. As a necessary consequence, our Tievs development must include execution of an internal process to detect fraudulent advertisements before publishing, so that a particular consumer is not swindled on advertising, or simply the user could have the privilege of progressing further simply by fixing it. As a result, the consumer would be more appreciative and keener to continue and then use the applications eagerly and effectively without doubting, rather than departing from it down to a single yet inconvenient issue, such as determining whether the advertising is legitimate on their own. Correspondingly the developed application would be established and constructed to accommodate multiple variations of advertisements, with subsequent updates surpassing all requirements, we have opted to confine our study due to the scope of this project, which should be completed within a year. This only pertains to vehicle (Car) advertisements. The descriptions of the advertisements of the vehicles were assessed in this approach since this has been considered as the most dominant.

The detection will be constructed by employing the yet most optimized machine learning model invented out of seven comprehensive ML algorithms and the detailed confluence of factors parameters and training the models, that will inevitably result in an optimum trained model, when utilizing the ‘Used Cars Dataset' acquired from Kaggle, where it contains descriptions and other attributes of 450,000 US used car detailed information that has been revised over the last three months by Austin Reese. The remainder of this work is organized in the following manner. The following section provides background information on the fraudulent activity on classified ad platforms. After discussing the techniques that fraudsters commonly exploit on classified ad sites, the problem of the study and gap analysis will be properly addressed and discussed the research's motivation. Section 2 discusses the design methodology, high-level architecture diagram, and major decisions for the application we designed. Section 3 describes the empirical impact of the application, the evaluation techniques, and the findings of the testing, and the proposed future works that will be disused. ultimately, the conclusion of the pape

1.2 Literature Survey

Even though it is distinctly apparent that online fraud detection has already been implemented by numerous parties in recent years, there is yet to be a highly ameliorated application that incorporates complex machine learning technology to enhance the prominent functionalities to facilitate customer expectations and requirements efficiently and there have been quite spectacular, enlightening outcomes in terms its effectiveness and efficiency aspects of machine learning techniques that have already been utilized for forecasting. But previous studies distantly allied with the car classifieds content fraud detection component is being addressed due to insufficient current research precisely covering the entire fraud detection solution. Over the years, researchers have established several techniques to detect online spam. supervised learning is the most common existence of these studies. Moreover, In the past literature, two categories of features have been utilized to distinguish spam and non-spam studies so far, either using link-based features or content-based features as n-grams. However, the Link-based feature does not support it appropriately since most of the time it is rarely linked to web pages together in advertisements.

Ntoulas et al [8] extracted features from web pages and content that can offer information to differentiate a spam web page from the regular ones. Using a real-world, dataset of more than 105 million web pages downloaded by the MSN Search crawler, they investigated several aspects of content-based spamming on the web. In a sample of 17,168 pages drawn from English pages, 86.2% are non-spam and 13.6% are spam A variety of heuristic approaches for identifying content-based spammers have already been presented. A few of their spam detection techniques are more successful than others; nonetheless, when employed alone, these methods may fail to detect all spam websites. As a consequence, they merged existing spam-detection algorithms to build a C4.5 classifier that is highly accurate. This classification accurately identifies 86.2 % of all scam webpages while incorrectly classifying relatively few legal pages as spam. A few of the spammer detection techniques discussed in this research may be readily tricked by fraudsters. For instance, maybe misled by the inclusion of common terms in the otherwise pointless text. Despite the anticipated that it would be difficult for a spammer web page to deceive all of their approaches, they expect to observe some decline in categorization performance and reliability. To compensate for this, they intended to investigate how they could well utilize NLP methods to detect artificially created text.

McCormick and W. Eberle [4] This study provided a technique for detecting fraud in advertising by employing well-known ml algorithms including Naive Bayes, ANNs, and Random Forest. Their original data mining strategy performed similarly to conventional techniques of others employ detection. A collection of classified ads advertising was used to test our method of identifying fraud in them. A company offered advertising and related data to a business that runs an online classified ad website

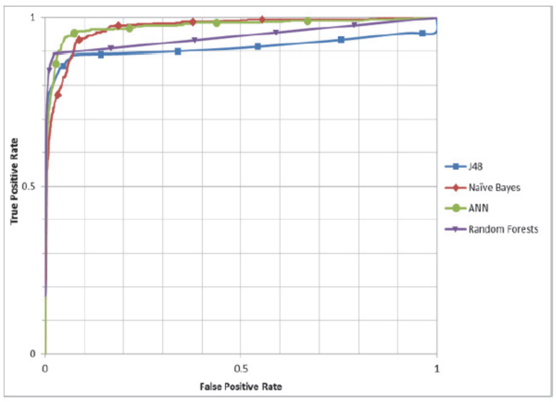


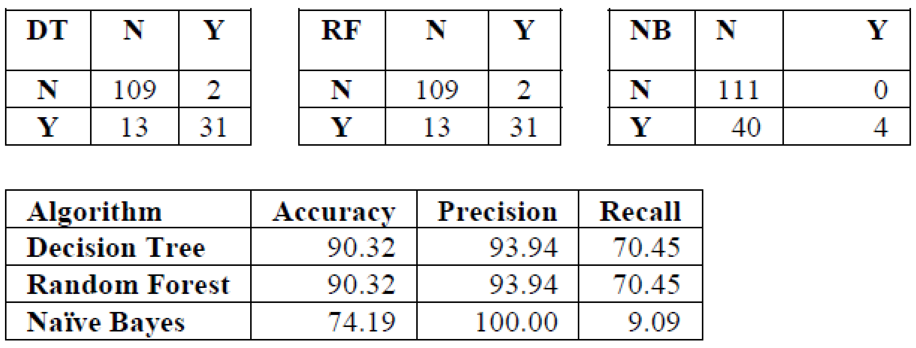
Figure 1.3 ROC graph showing performance of classifiers

Furthermore, when evaluating the overall effectiveness of the assessed Random forests serve as classifier provided existing data to be the proper approach, with a 0.969 ROC area. Whereas the recall rate of ANNs ultimately approaches that of random forests, that is not till the accuracy of the classifier undergoes a significant decline. Employing random forests may indeed be offered a piece of extra information. by investigating the relevance of each characteristic the produced decision trees will assist in selecting an upcoming feature extraction. The above ROC graph Figure 1-3 was used to evaluate how each method in their study might be modified concerning this trade-off.

A research study conducted by Hung Tran et al [1] introduced a technique for detecting spam in advertisement posts. Towards the authors ’ knowledge, their research constitutes an endeavor at spam identification within that critical domain. Firstly, they analyze specific attributes of that kind of sector in contrast to other domains and investigate why prior methodologies are inadequate for the online marketing sector. Furthermore, they provide a novel set of domain-specific characteristics for distinguishing spam from non-spam advertising posts. Finally, researchers undertook tests using a Free classified dataset to demonstrate the feature set's efficacy. In this work, In terms of precision and recall, their method outperforms the baseline utilizing only conventional n-gram features by 59 % and 52 %, respectively. In terms of F-1 overall performance, their method outperforms the baseline by 55%. Researchers intended to expand the testing dataset in future employment. Finding appropriate cases for assessment is a fascinating challenge in and of itself. Because it was such a tremendously unbalanced classification problem, there were relatively fewer spam cases in the population if they randomly select a sample of examples for judgment. To address this, researchers want to employ the active learning technique in future studies.

The approach provided by the author [9] utilized term frequency and inverse document frequency representation for messages, clustering a sample of messages from the training pool and deriving labels for cluster medoids. Establishing an initial Random Forest model for classification and prediction; acquiring labels for further training instances varies depending on unpredictability, and fine-tuning the Random Forest model. The capability of clustering prototypes to efficiently represent compact, properly separated clusters is the most unique discovery from this done by authors. A burst of 2,532 messages from April 13th, for example (more than 25% of the training pool), can be adequately associated with a specific prototype. This cluster was a spam effort masquerading as a vital advisory. Another instance of a compact, but also well clustered was a pharmaceutical product advertising represented in their study. Cluster prototypes were another relevant depiction of the Random Forest algorithm's development pool. Considering an initial selection of messages for classification based on cluster prototype, followed by selecting an extra batch of messages for model enhancement based on confusion and uncertainty, improves performance. Accuracy is tremendously competitive with formerly published discoveries with as less as 100 classified content from one week of data. Their next step was to choose messages for classification based on the appearance of new groupings in the stream of data for just a deployed model. Its representation was also be updated to incorporate information about the messages' links and pictures.

The objective of the study constructed by Suresh Yaram [10] is to emphasize the key aspects of machine learning approaches such as Documents Classification and Prediction using algorithms such as Random Forest and Nave Bayes. for prevention and detection of fraud. Implementation of these methods and techniques, along with appropriate data, have indeed been explored. Statistical modeling tool 'R' has been used to establish industrial use cases. The purpose of presenting this study was to demonstrate the aspects of prominent machine learning approaches and consolidate each one together use of these algorithms throughout the industry verticals. Data for this investigation were collected and analyzed from a variety of sources, encompassing structured semi-structured (CSV), unstructured (document, images, audio/video streams), and semi-structured (XML). Furthermore, the confusion matrix was used to make a comparison of three classification methods which allows computing performance indicators such as "accuracy," "precision," and "recall".A total of 500 observational data were arbitrarily partitioned into training and testing data (70:30). Decision Tree and Random Forest algorithms consistently outperform Naive Bayes in terms of accuracy. According to research observations, Decision Tree and Random Forest are the best options which accomplished the accuracy of 90.32% as shown below fig.



DT	N	Y
N	109	2
Y	13	31

RF	N	Y
N	109	2
Y	13	31

NB	N	Y
N	111	0
Y	40	4

Algorithm	Accuracy	Precision	Recall
Decision Tree	90.32	93.94	70.45
Random Forest	90.32	93.94	70.45
Naïve Bayes	74.19	100.00	9.09

Figure 1.4 Accuracy accomplished by Research 10

1.3 Research Gap

Various approaches for detecting online malicious content have indeed been initiated throughout the years by research scientists. The foremost prevalent field of research is based on machine learning. Conversely, in the literature review, a variety of characteristics were being used to distinguish inappropriate content from non-spam. Even though certain research studies had already endeavored to construct splendid models adept at scrutinizing online fraudulent behavior of advertising products with adequate accuracy and robustness, much more untapped novel approaches have yet to be accomplished.

Most of the research that was conducted before has used only a small number of records as their datasets. For example, [8] and [10] conceded data to trainand analyze the models which having extremely below the amount that this proposed component would be working on, which is about 450,000 records (after preprocessing can be reduced but slightly). Hence the issue of models being biased to training data will be reduced since the rangeof records is higher and will additionally improve the prediction accuracy when thereare more samples as well.

When considering the available research papers [1],[4],[8],[9],[10], it was vividly noticeable the authors have never used LightGBM classifier which is a successful predictive model itself having the speed faster than other similar performing solutions in the field, flattering scalability, and low memory usage, when compared with other boosting algorithms. For fraud content analysis.

Nevertheless, [8], [1],[10] have used the traditional algorithms, whereas in this study more powerful boosting algorithms were compared in comparative analysis and evaluated using a classification matrix. Likewise, there are fewer researches prevailing that have considered a dataset with a high number of records, and none of them have considered LightGBM classifier for detecting frauds, while most considered individual algorithms for detection.

Hence, I have decided to take into account LightGBM, and six other machine learning algorithms having two boosting and others identified through the Literature review, having the excellent capability, make the necessary refinements, analyze individual model accuracy and implement an optimum model to detect fraud, considering composite fidelity and time complexity.

When analyzing the research papers further [1] - [5], [8]-[10], it is understandable to the reader that the authors have analyzed and investigated the performance, accuracy, and error levels, for only a single machine learning algorithm, in the context of fraud detection. Therefore, the proposed system was implementing an optimum model while the proper hyperparameter tweaking process was undertaken which has never been attempted as in previously fulfilled explorations.

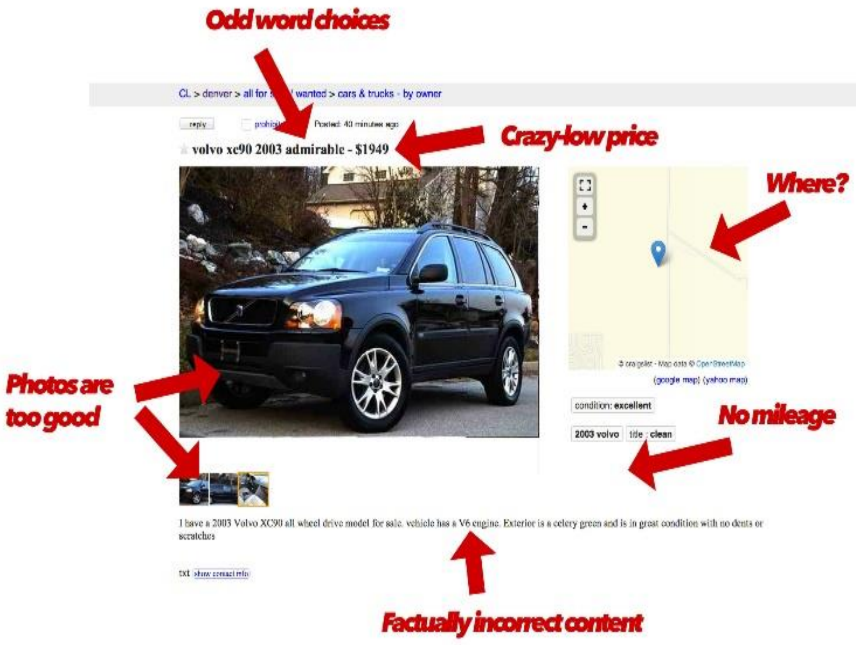
Many types of research [5]-[8], [10]-[11] have not given attention to the classified's fraud using content analysis based on used car classifieds. Therefore, As concluding this study was undertaken to perform specific techniques used preprocessing, exploratory data analysis, and comparative analysis among seven algorithms with parameter optimization while the most appropriate approach for fraud content analysis was revealed having a complex dataset with a higher number of records. Table 1-1 below depicts in tabular form, the summarization of the above explanation.

Table 1.1 Comparison of existing research studies

Fact \ Research	[1]	[4]	[8]	[9]	[10]	Proposed System
A complex dataset with an enormous number of records	✓	✓	✓	✗	✗	✓
Inclusion of LightGBM model in fraud detection	✗	✗	✗	✗	✗	✓
Implementation of the optimum model premised on a comparative analysis of seven algorithms and three boosting procedures	✗	✗	✗	✗	✗	✓
Consideration of car classified advertisements	✓	✗	✓	✗	✗	✓
Model optimization using hyperparameter tweaking	✗	✗	✓	✓	✗	✓

1.4 Research Problem

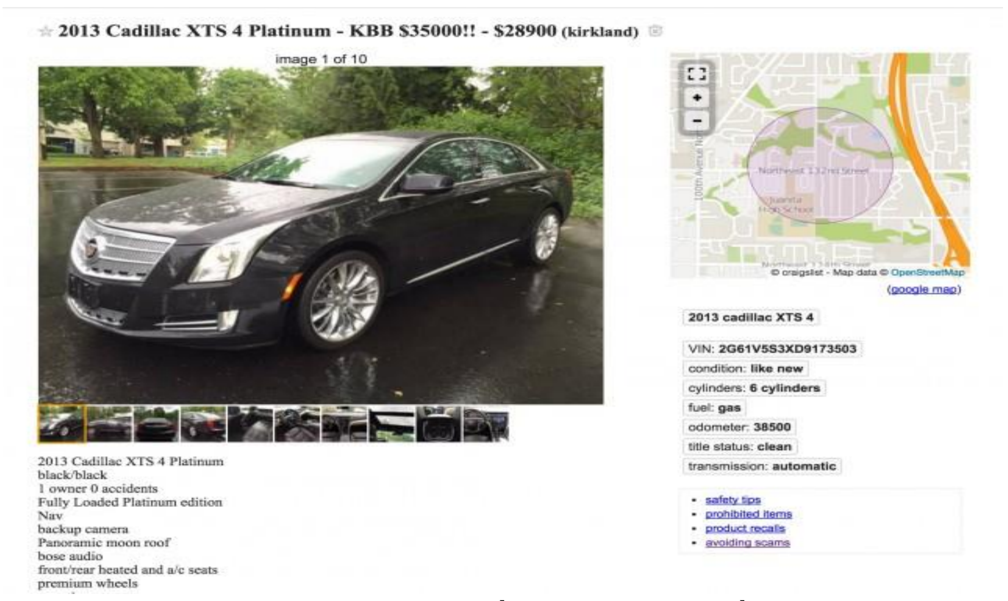
Even though it is distinctly apparent that online classified advertising platforms have already been implemented by numerous parties in recent years, there is yet to be a highly ameliorated application that incorporates complex machine learning technology to enhance the prominent functionalities to facilitate customer expectations and requirements efficiently. Moreover, these online advertisements have become the most essential part of the advertising market since the development of the internet. And those online classified advertisement sites have a huge number of advertisements daily (Eg: eBay, Craigslist). Though the popularity of the online advertisement sector, it was highly targeted by spammers. The main problem is most classified advertisement sites **do not give the priority** to, **Identifying and preventing fraudulent advertisements or inappropriate content prior to submitting**. Thebelow Figure 1-5 shows a real-world example of a spam post, acquired from Craigslist Classifieds while Figure 1-6 depicts the real-world benign posts. Hence the proactively recognizing, detecting fraudulent advertisement content information, preventing it from being posted in the application, is a crucial problem at present. The widespread accessibility of the web has the unwanted effect of attracting online scammers who pose as genuine sellers by posting fake advertisements to defraud would-be buyers.



volvo xc90 2003 admirable - $1949

I have a 2003 Volvo XC90 all wheel drive model for sale. vehicle has a V6 engine. Exterior is a celery green and is in great condition with no dents or scratches

Figure 1.5 Scammer Advertisement on Craigslist



☆ 2013 Cadillac XTS 4 Platinum - KBB $35000!! - $28900 (kirkland)

image 1 of 10

2013 Cadillac XTS 4 Platinum
black/black
1 owner 0 accidents
Fully Loaded Platinum edition
Nav
backup camera
Panoramic moon roof
bose audio
front/rear heated and a/c seats
premium wheels

© craigslist - Map data © OpenStreetMap
(google map)

2013 cadillac XTS 4

VIN: 2G61V5S3XD9173503
condition: like new
cylinders: 6 cylinders
fuel: gas
odometer: 38500
title status: clean
transmission: automatic

- safety tips
- prohibited items
- product recalls
- avoiding scams

Figure 1.6 Legitimate Advertisement on Craigslist

Scammers can steal millions of dollars from unsuspecting users and threaten the reputation and utility of online ad services [4]. To solve the issue, researchers conducted investigations such as [1]-[10] to create machine learning models with the appropriate intelligence and effectiveness for detecting fraud content in used vehicle advertising. Nevertheless, while reviewing the study, several machine learning methods created from clustering or classification procedures have already been thoroughly studied, and the findings obtained have been possible to deal with intricate circumstances.

From the applications' developers and owners' point of view, they must implement strict program logic to identify or prevent fraudulent advertisements from being posted (fake spam/ irrelevant descriptions) accidentally or deliberately by users of the system, through verification and validation processes internally. This will require inspecting posts separately and meticulously, even after the ad being posted to the customers and if so, customers will view those fraud ads and lose confidence in using the platform for their needs, enabling loss of application reputation as well [2].

Most classified advertising products currently in use do not prioritize detecting and preventing fake advertisements or characteristic details from being submitted, let alone inspect them internally and inform the respected users who are in the process of submitting the advertisement. Nevertheless, many models of machine learning constructed either by regression or classification processes were inventively analyzed by the research, and the accuracy obtained in some cases was successful, while many others did not demonstrate commendable recognition. Despite the fact that some were not appreciative. Many of them seem to have not examined complicated datasets with a huge number of records for training and validation of the algorithms, even though additional data will assuredly increase the model's inaccuracies. Analysis of ad characteristics that increase the chance of spamming while retaining the model's straightforwardness, which has not been adequately researched by most scholars.

Performance, as precision, is an important factor in forecasting, even though the LightGBM classifier model, which has the topmost range in speediness as according, has not yet been examined in prior experiments for fraudulent content detection, which is indicating a research gap. Even after the ad is posted, customers will perceive those fraudulent ads, end up losing confidence in using the platform for their necessities, and enabling loss of application name and reputation, although many types of research, such as [1-9], have not adequately argued the significance, resulting in a major misplacement for persistent fraud content detection system.

As a result, the suggested fraud content detection system seeks to solve the prevalent difficulties of incomparable research studies, as described above, and sufficiently make novel changes as solutions to improving the user experience for the application's consumers.

1.5 Main Objective

- The predominant objective is proactively identifying and **preventing fraudulent advertisement content information, from being posted** in the application, using machine learning appliances

  This study incorporates a machine-learning algorithm to perform a counterfeit/fraudulent advertising detection and classification system to preserve the advertising platform from being spammed by spammers or consumers with misleading behaviors who have been in the process of submitting an advert for an item of a particular variety (such as Cars) throughout the year. As a result, the predominant objective is to implement machine learning appliances to detect and prevent fraudulent advertising content information from being published in the app.

  An ameliorated, systematic, and rich user-friendly solution to enable interested parties to post their advertisements efficiently and accurately without any inconveniences, in addition to providing an optimized, interactive search facility within the system, to search and find items to buy, according to specific customer requirements, seamlessly and productively moreover, design and develop an internal process, using machine learning appliances, to proactively identify fraudulent advertisement information context having inappropriate/irrelevant content being uploaded into the application. (Will be targeting one type of classified such as vehicles) . When a customer is about to submit their advertisement(s) into the system, proactive inspections will be done through the system to find out whether the current advertisement is being inspected consisting of spam or inappropriate content information. If so, the customer will be prompted with a warning message, indicating such matters present in the advertisement, which is yet to be submitted, so that the customer could proceed with the necessary measures to be taken to avoid such schemes. Nevertheless, because of its scope and time limitations imposed on the system, these functional requirements are implemented and aim at one kind of classified car. The emphasis will therefore be on identifying and presenting, according to requirements described by consumers on the advertised form they fill before submissions, fraudulent advertisements of the used or second-hand vehicle.

1.6 Specific Objectives

To reach the concept of proactively identifying and preventing fraudulent advertisement content information, from being posted in the application, using machine learning appliances the specific objectives that need to be accomplished is as listed below,

- Data acquisition

  Congregate the appropriate data set using external sources (e.g.car_ad.csv/ vehicles.csv), or survey using craigslist region to train the model and test the model built and initial data records should be labeled as fraud and non-fraud in the ability to predict the likelihood of counterfeit content,

- Explore the appropriate algorithm

  Explore the most suitable machine learning algorithm considering the accurateness and / interpretability of the output speed or training time, linearity, etc., along with seven algorithms with boosting methods evaluation in comparative analysis and exposed the optimum model compared and hyperparameter tweaking.

- Develop and train the model

  Develop and train the model to detect, whether the advertisement is being inspected by inappropriate /irrelevant content exploitation techniques as exploratory data analysis and data preprocessing

- Integration of the developed model

  Fraud content detection Flask API implementation, Embedding the Mode importing the model pickles with the application without inconveniences.

- Build the client and server components

  Build client and server-side components and extract the data from the advertisement form, before posting. If the advertisement is consisting of inappropriate content, the client will be prompted with a warning message, indicating such issues present in the advertisement, which is yet to be submitted

2 METHODOLOGY

2.1 Methodology

2.1.1 Data preparation high-Level system diagram

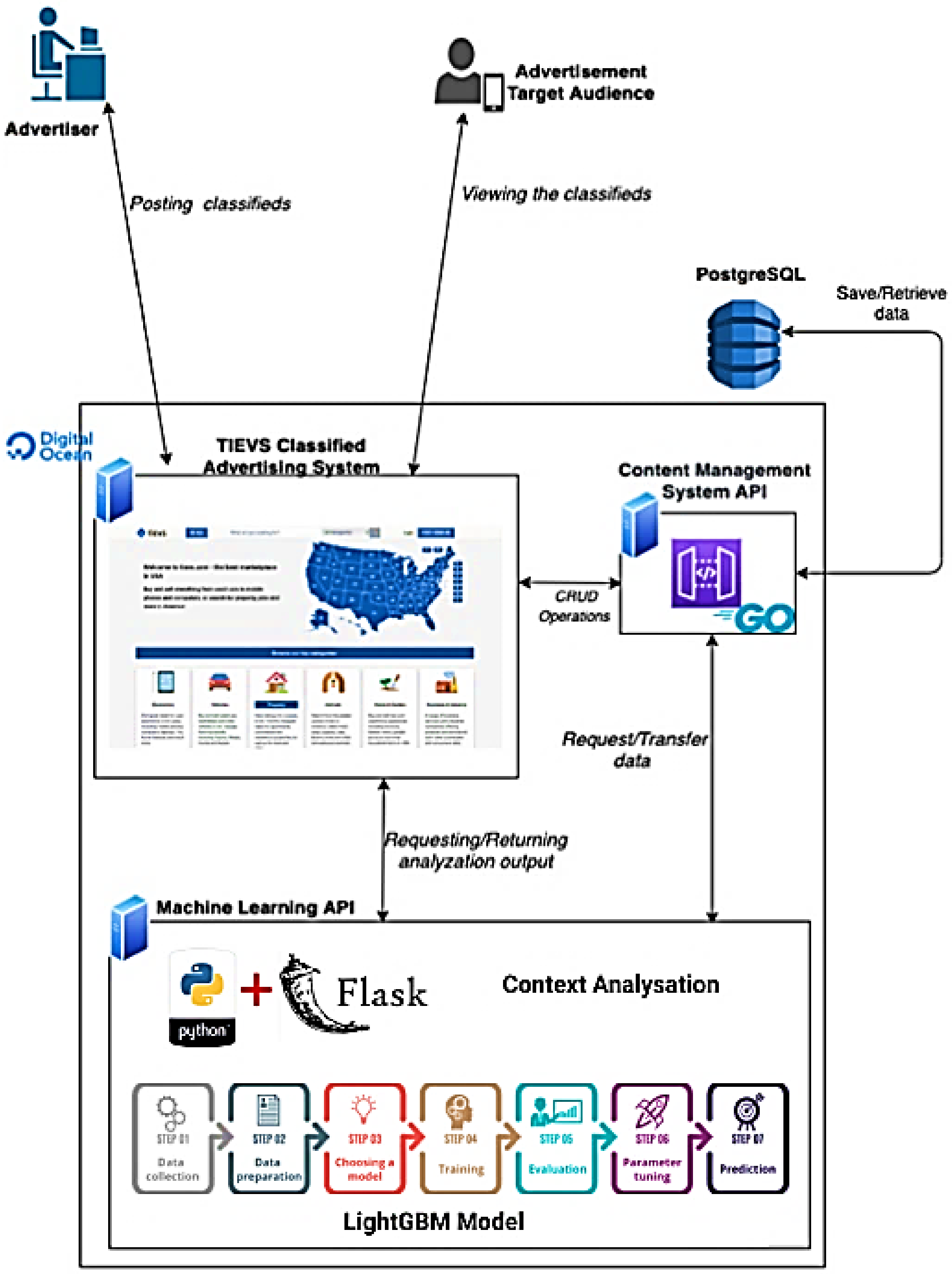


Figure 2.1 Research Methodology Diagram

The prominent characteristic of “Tievs” is that it offers the customer to publish their own advertising to the public in an effortlessly offering or purchasing matter in a convenient and smooth manner. As stated in the research question above, detecting and preventing fake advertising from being published, whether accidentally or deliberately, employing verification and validation approaches is a tremendous undertaking. And furthermore, in accordance with the most recent rising technological discoveries, exhibit rich User Interfaces (UIs) for seamless user experience or enhance excellent customer experience. Therefore, a method for detecting fraud context on online classifieds based on Machine Learning is developed. This technique for spotting misleading/fraudulent content will be constructed by obtaining the inputs as text content/description, labeling them as benign or malignant, and then assessing the findings to produce sufficient the best-generalized output. Using these instances figures mentioned above research problem, the created system will be able to understand how postings in a specific category appear. Once the approach is implemented, we will be able to identify potential postings as spam or not prior to uploading them to the application. For the optimum functionality of the above-described system, the next stated aspects,which are rigorously emphasized, should be thoroughly realized and attained.

The entire high-level system architecture of the newest component architecture built for car classifieds fraud prevention and detection is illustrated in Figure 2-1. The Flask machine learning API would first extract the dataset from the database, where it may be aggregated and saved using a content management system. Following the acquisition of the dataset, data exploration and different data preparation procedures will be used to purify the dataset before it is transferred into the algorithms. After the dataset has been properly prepared, it will be divided into subsets and put into the machine learning algorithm model that is thought to be the best for this study concept. The description will be taken from the advertising form in the application interface into the flask API, which has the model, and the detection result will be generated after rigorous inspection of the characteristics. then, Following the model's accuracy, it will be integrated also with the Angular application to identify fraudulent contents of car advertising that engage with the Frontend. The identification will be exhibited to the consumer as a fraud notice within the advertising form, near the descriptions input section. If a legitimate user posts, it will be stored in the database whereas the user permits the submission to proceed. These contents are persistently dynamic and might vary from one advertisement to the next. Consequently, the primary goal of this aspect is to undertake supervised learning and determine the likelihood of spam based on the inputs. Below offers a brief summarization of the above high-level procedures.

- Aiming to foresee the likelihood of fraudulent content prior to ad submission, an initial dataset record was labeled as legitimate and illegitimate, where the latter was prepared.

- Afterward successfully installing all of the packages (including libraries and dependencies) required for the procedure afterward to obtain a communicative impression in the forms of charts and graphs the exploratory data analysis (EDA) technique was undertaken primarily.

- Succeeding EDA, preprocess technique, which is discussed in the implementation section, often perceived in writing regardless of context was utilized to eliminate all interference from inputs that would otherwise obstruct the potential to distinguish positive and negative classifiers.

- The procedure of the Stemmer algorithm, which eliminates affixes to retrieve the basic form of words, was utilized as a stemming technique due to its computational speed.

- The raw text was essentially converted into several vectors to perform in ML models due to its impotence to directly operate textual forms.

- Then, a Bag of Words (BOW) model was developed to extract characteristics from the text before the implementation of the TF-IDF NLP vectorization technique to evaluate the significance of text data and serialize them for further transformation. Subsequently, the Scikit-learn library was utilized to convert text content into a matrix of numbers that could be fed to the classifier.

- This study was initiated to evaluate the expertise of 7 ML algorithms such as Decision Tree, RF, SVM, KNN, XGBoost, XGBRF, and Light Gradient Boosting Machine (LightGBM) classifier in a comparative analysis after vectorization. This is the development's machine learning component. Our current issue is determining if a user is going to submit a post as spam or not. This is a conventional machine learning classification issue in which one class is ham (positive class) and the other is spam (positive class). Our objective is to be able to create a system that can identify whether a user will submit a posting as spam.

- One strategy to solving this challenge is to utilize supervised learning and is what we have been using in our research. In this technique, there is indeed a learning phase in which our algorithm "gets to know" about the features of spamming and innocuous postings after being fed certain tagged posts.

- Each classifier was trained and tested until the optimal model would be exposed in terms of F1 score (accuracy and recall) and confusion matrix.

- Based on iteration, proceeding with LightGBM as optimum model disclosure, the highest max depth value was identified from parameter tuning compared with competency to accomplish the most pertinent parameters for the best classifier.

- After successful embedding of the trained model in the REST API fraud-detection endpoint, it could classify ad content as legitimate or not and display caution Tievs near advertisement submission when indispensable.

2.1.2 Technologies and tools used

As technological tools, Jupyter Notebook, Anaconda Navigator, Visual Studio Code, and PyCharm were utilized, while PostgreSQL was used as the database server. Also, Postman to test REST APIs was used and the Supervisor system was used for resuscitating APIs in the event of a hosting malfunction. Python (Flask/Quart), Angular (Typescript), and supporting libraries such as Pandas, NumPy, Scikit-learn, Angular Material, and others were used to create a Classification model, front-end apps, and model/Content Management System APIs.

Figure 2.2 Technologies used as developing

2.1.3 Software development life cycle

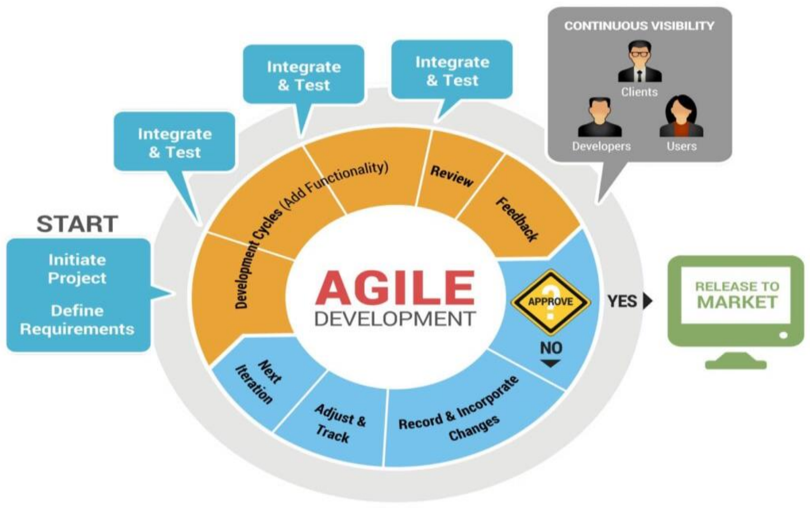


Figure 2.3 Agile Methodology

Agility is the ability to respond to unpredictable changes with quick response and profitability [15]. Many companies benefit from enhanced job organization, cooperation with industry, and predictability of performance metrics. This effort will be led during the year by the notorious agile development and constant improvement aspirated, change inviting, Agile methodology strategy, as shown in Fig 2.1.2 Moreover, Agile principles help increase the growth of almost any implementation since these processes are versatile and the rules do not have to be thoroughly understood.

Although a set plan is constructed to aid for the time management and delivery schedule alignments with the help of a Gantt chart, according to the Agile values, if impediments were to occur, the necessary adaptations must be taken into place in a timely manner, without interrupting the continuous development processes [16]. At these kinds of moments, reacting to change efficiently and effectively, as well as ensuring the consistency of the project timeline, are critical. As the Agile approach is used throughout the software development cycle, that would be much simpler to consider the challenges by turning them into opportunities for success. Ultimately, Scrum, a lightweight edition of the massive Agile framework, can be used to accommodate high-quality incremental goals and objectives in short durations, as well as team performance and application development.

2.1.4 Project requirements

- Functional requirements

  - Successful model implementation using seven different algorithms with three boosting algorithms and obtained the most suitable classifier to further performed.
  - Successfully obtained the optimum model for fraud context detection utilizing the hyperparameter tweaking along with the randomizedGridSearchCv.
  - Proper client-server integration (Flask fraud detection API – "Tievs" Angular front-end).
  - Flawless cloud deployment of front end and flask API and without any inconsistencies.
  - Successful customer navigation to application and the proper visual interface of advertising process elements should be implemented for a convenient representation of the different features of a car.
  - Extraction of accurate data to establish the fraud detection model.
  - The 'Submit' button should preserve its precise functionality. And Finally, proactively identify and prompt a warning message prior to submitting the advertisement.

- Non - Functional requirements

  - Accuracy
  - Security
  - Ease of use
  - Adaptability
  - Scalability
  - Efficiency
  - Ease of use
  - Timesaving
  - Accessibility
  - Maintainability

2.2 Commercialization Aspects Of The Product

Fraudulent activities have infiltrated nearly every aspect of online business, including cybersecurity incidents that threaten end consumers' privacy rights and e-commerce transactions to ransom assaults that extort large amounts of money from businesses. When fraud is not addressed proactively, it may have a detrimental impact on a firm's profitability by diverting money away from core business and priorities, inflicting brand degradation to its reputation, and dissipating revenues. It could even cost people their lives in extreme circumstances. As ours, to seek out new tactics and combat fraud, modern fraud prevention and detection platforms employ a comprehensive system that incorporates ML, analytics, and other techniques. The amalgamation of a proactively identified fraud detection system together within a web-based car classification platform could perhaps lead to significantly greater and convenient revenues of the advertised vehicles since such a web application has an elevated prestige and authenticity, which would be more advantageous and authentic when especially in comparison to other equivalent cars available. Consumers wouldn't even need to undertake considerable investigation while determining the trustworthiness of a classified ad by quitting the application. Clients would achieve based on a genuine selling feed from the program itself which would be completely satisfied with the response that spared clients valuable time from their hectic schedules and will focus on providing it a huge competitive advantage over all the other equivalent platforms that make identical efforts to accommodate customers. Because of all these scenarios and the high interactions with clients and utilization, the platform's commercial revenue will increase as well.

Figure 2.4 Commercialization aspect of the system

2.3 Fraud Detection System Implementation

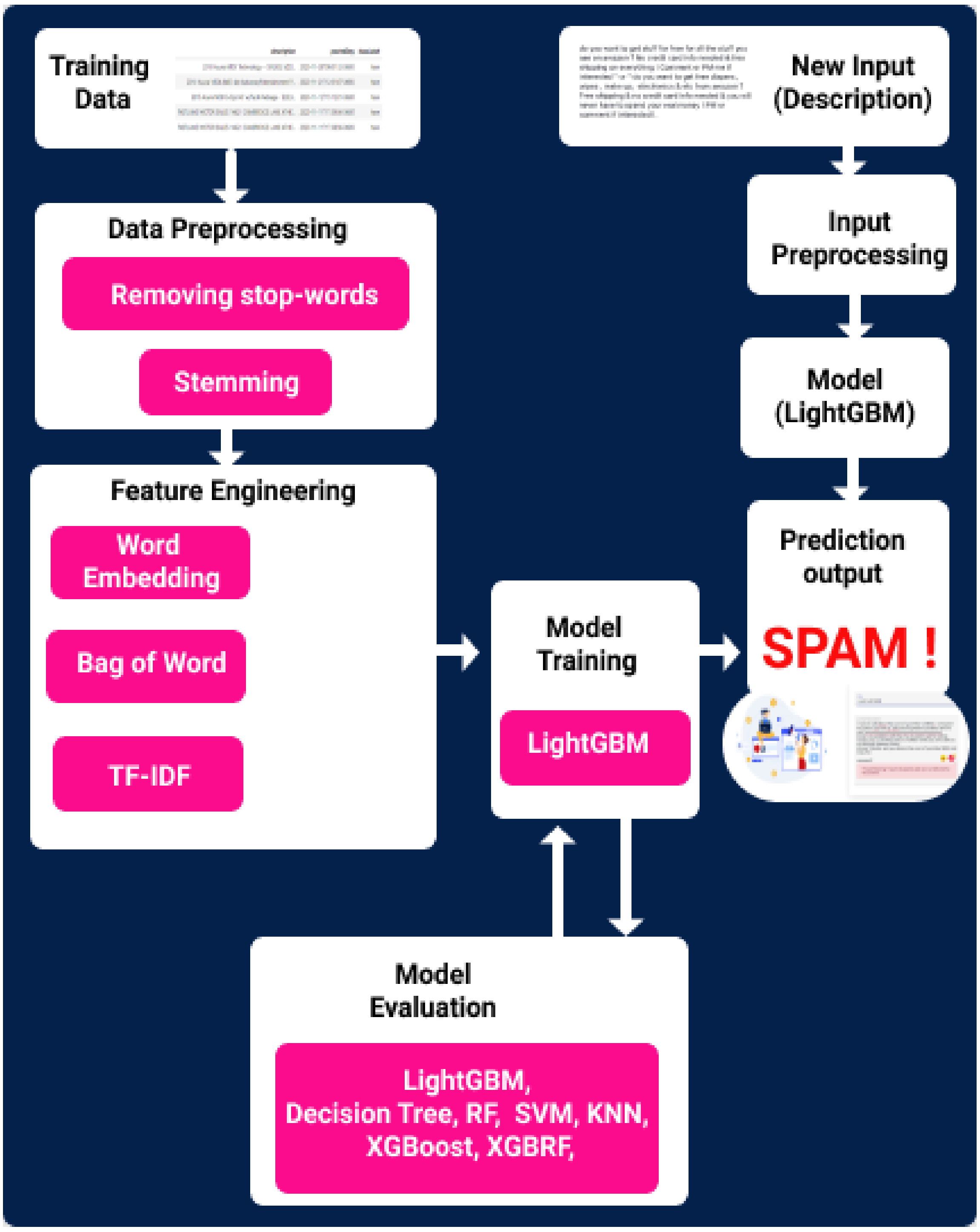
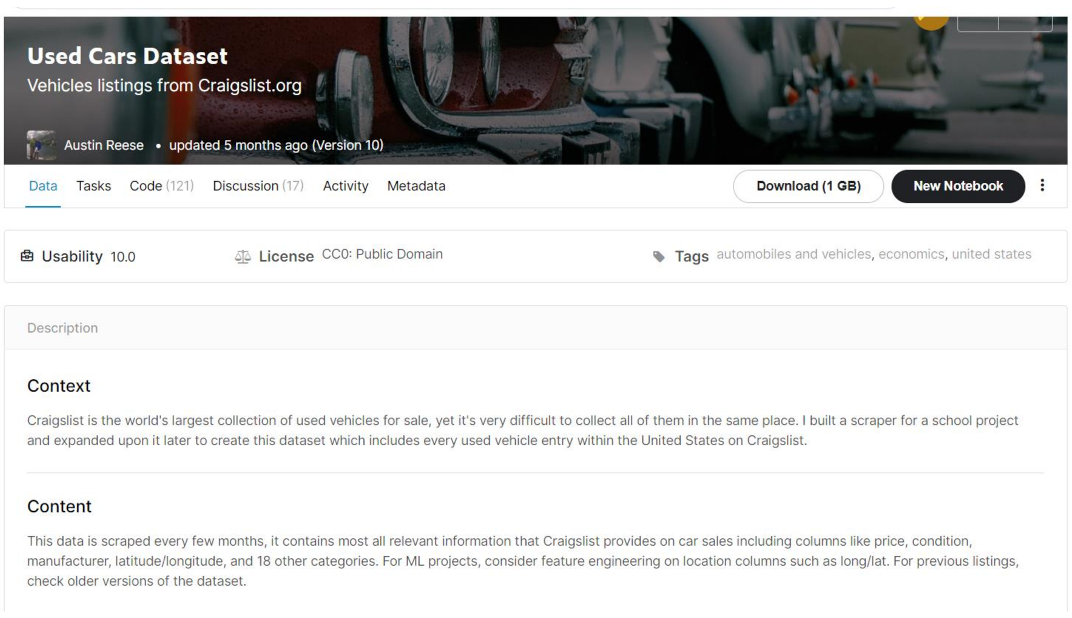


Figure 2.5 Process diagram of the developed system

3.3.1 Data Acquisition Phase

For the required ML model rudiments, A preliminary dataset of 23,000 records was labeled as genuine and illegitimate inability to forecast the likelihood of counterfeit content prior to ad submission, with the latter provided in addition readily accessible 'Used Cars Dataset' Kaggle datasets were being used. Austin Reese initially disclosed and revised the aforementioned dataset, which was obtained from one of the most commonly available online classified advertisement platforms, Craigslist.com, which is barely acknowledged and is accessed by many consumers in the United States. The dataset incorporates 441,396 used car records and encompasses about 40 brands with 25 unique attributes which also be excluded due to inconclusiveness or empty fields. Since 'Tievs' initially targeted the US market, the dataset was obtained two months before the outset of this research from an American classified advertisement portal, called 'Craigslist'. This precise dataset is appropriate for the development of the fraud forecasting component because the proposed system is basically intended to target US customers. Consequently, info concerning used cars in the United States is pertinent to the advancement of this matter.

Dataset Web Link - https://www.kaggle.com/austinreese/craigslist-carstrucks-data


Used Cars Dataset
Vehicles listings from Craigslist.org
Austin Reese • updated 5 months ago (Version 10)
Data
Tasks
Code (121)
Discussion (17)
Activity
Metadata
Download (1 GB)
New Notebook
Usability 10.0
License CC0: Public Domain
Tags automobiles and vehicles, economics, united states
Description

Context

Craigslist is the world's largest collection of used vehicles for sale, yet it's very difficult to collect all of them in the same place. I built a scraper for a school project and expanded upon it later to create this dataset which includes every used vehicle entry within the United States on Craigslist.

Content

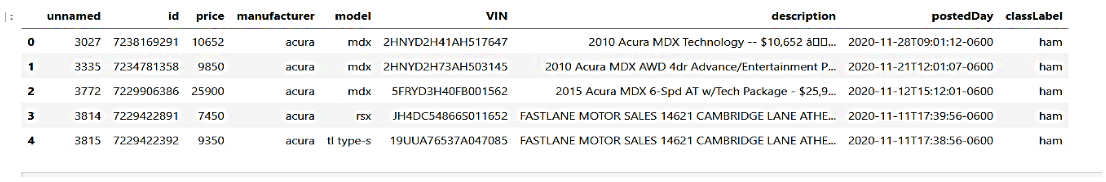
This data is scraped every few months, it contains most all relevant information that Craigslist provides on car sales including columns like price, condition, manufacturer, latitude/longitude, and 18 other categories. For ML projects, consider feature engineering on location columns such as long/lat. For previous listings, check older versions of the dataset.

Figure 2.6 Used car dataset utilized from Kaggle

3.3.2 Data preparation

- **Exploratory data analysis**

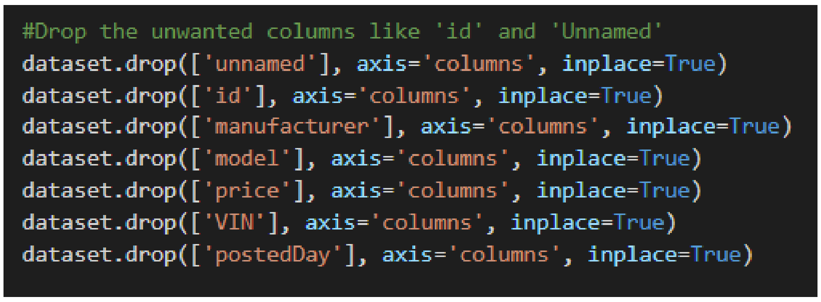
Exploratory Data Analysis (EDA) pertains to the crucial operation of conducting preliminary studies on data in order to distinguish correlations, spot anomalies, test hypotheses, and validate assumptions employing statistical results and graphical representations. It is advisable to first comprehend the data and then strive to glean as many insights as possible from it. To perform EDA initially, the required libraries were imported, and the dataset was included. Below Fig 2.6 depicts an overview of the dataset.



	unnamed	id	price	manufacturer	model	VIN	description	postedDay	classLabel
0	3027	7238169291	10652	acura	mdx	2HNYD2H41AH517647	2010 Acura MDX Technology -- $10,652 â□□...	2020-11-28T09:01:12-0600	ham
1	3335	7234781358	9850	acura	mdx	2HNYD2H73AH503145	2010 Acura MDX AWD 4dr Advance/Entertainment P...	2020-11-21T12:01:07-0600	ham
2	3772	7229906386	25900	acura	mdx	5FRYD3H40FB001562	2015 Acura MDX 6-Spd AT w/Tech Package - $25,9...	2020-11-12T15:12:01-0600	ham
3	3814	7229422891	7450	acura	rsx	JH4DC54866S011652	FASTLANE MOTOR SALES 14621 CAMBRIDGE LANE ATHE...	2020-11-11T17:39:56-0600	ham
4	3815	7229422392	9350	acura	tl type-s	19UUA76537A047085	FASTLANE MOTOR SALES 14621 CAMBRIDGE LANE ATHE...	2020-11-11T17:38:56-0600	ham

Figure 2.7 Overview of the dataset

There were 5 entries in the dataset. Column v1 comprises the variable label ("ham" or "spam"), whereas column v2 includes the descriptive content. Columns "Unnamed: 1," include "NaN" (not a number) to simply indicate missing values and dropping unwanted columns as the depicted figure. They are not probably required; hence they can be eliminated as they would not be beneficial in attempting to create the model. To optimize the dataset's readability, the developed code snippet will eliminate and modify the columns.



```
#Drop the unwanted columns like 'id' and 'Unnamed'
dataset.drop(['unnamed'], axis='columns', inplace=True)
dataset.drop(['id'], axis='columns', inplace=True)
dataset.drop(['manufacturer'], axis='columns', inplace=True)
dataset.drop(['model'], axis='columns', inplace=True)
dataset.drop(['price'], axis='columns', inplace=True)
dataset.drop(['VIN'], axis='columns', inplace=True)
dataset.drop(['postedDay'], axis='columns', inplace=True)
```

Figure 2.8 Dropping the unwanted columns

Following that could do further feature extraction hence the length of the text may expose some technical insights. The dispersion of ham and spam text lengths is concentrated as below Fig 2.8 demonstrating that spam text is considerably shorter than spam ones

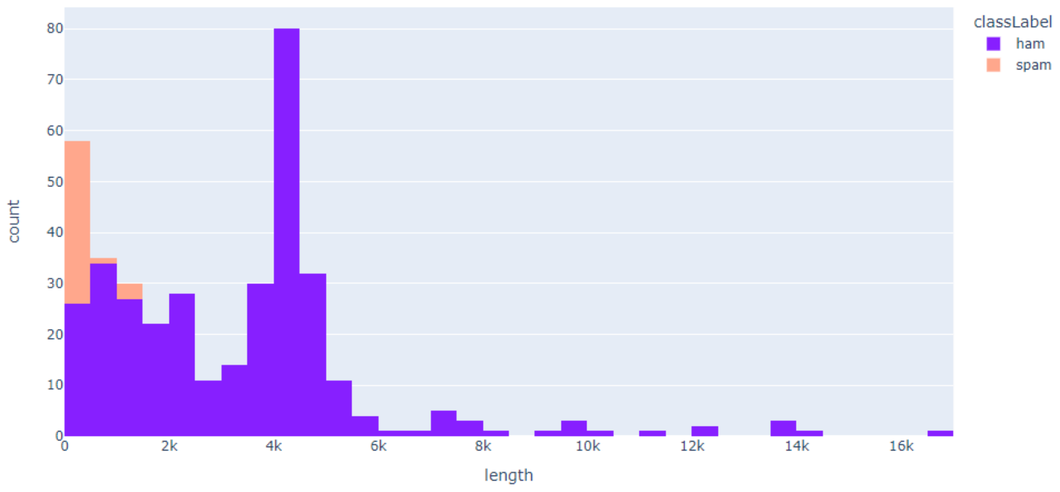


Figure 2.9 Length of descriptions

For a selected data segment spam accounts for 9.5%, whereas ham accounts for 90.5% can be shown as below Fig 2.9.

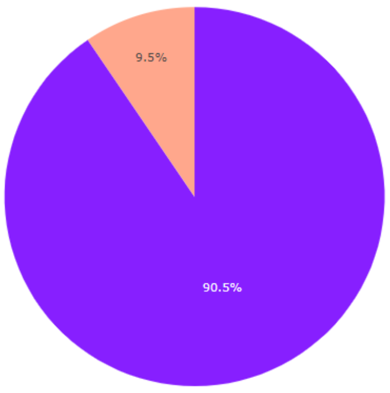


Figure 2.10 Spam / Ham distribution of selected data segment

Despite knowing, the highest prevalent terms used throughout spams and hams could assist us in properly comprehend the dataset. A word cloud assistance exemplifies exactly what sort of terms are prevalent within every classification. Initially, To provoke a word cloud, partition the categories into two panda's data frames and implement a basic word cloud procedure, as indicated below in Fig 2.10

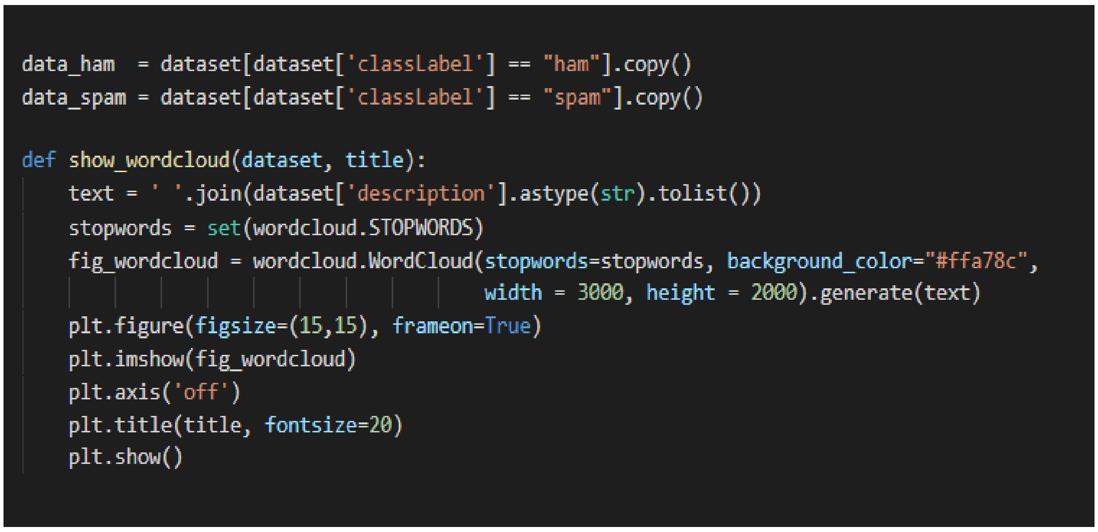

```
data_ham  = dataset[dataset['classLabel'] == "ham"].copy()
data_spam = dataset[dataset['classLabel'] == "spam"].copy()

def show_wordcloud(dataset, title):
    text = ' '.join(dataset['description'].astype(str).tolist())
    stopwords = set(wordcloud.STOPWORDS)
    fig_wordcloud = wordcloud.WordCloud(stopwords=stopwords, background_color="#ffa78c",
                                        width = 3000, height = 2000).generate(text)
    plt.figure(figsize=(15,15), frameon=True)
    plt.imshow(fig_wordcloud)
    plt.axis('off')
    plt.title(title, fontsize=20)
    plt.show()
```

Figure 2.11 Visualization of "ham" and "spam " word cloud

- **Data Preprocessing**

Data preprocessing is an indispensable phase in Machine Learning since the integrity of information and the crucial assets that can be extracted from it directly affect the model's potential to perform; hence, it is imperative that we preprocess our data prior to feeding it into our model. This incorporates methodological approaches for processing text content for the proposed machine learning model.

The concepts that will cover.

1. Label encoding
2. Handling null values
3. Regular expressions
4. Handling categorical variables
5. Removing stop words and stemming
6. NLP technique TF-IDF
7. Bag of words

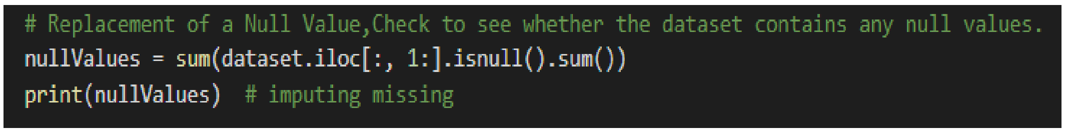
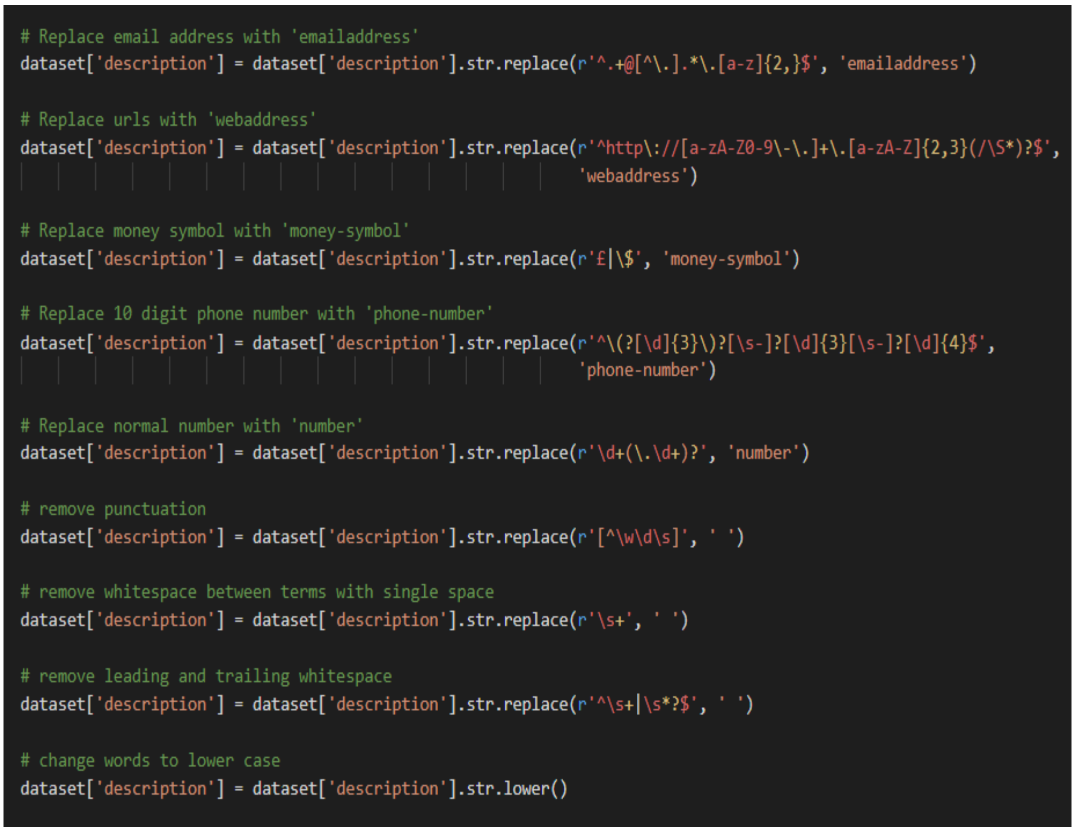

```
# Replacement of a Null Value,Check to see whether the dataset contains any null values.
nullValues = sum(dataset.iloc[:, 1:].isnull().sum())
print(nullValues)  # imputing missing
```

Figure 2.12 Replacement of null values

Initially, the Replacement of null values is done as shown above in Fig 2.11. Converting the label to a numerical is required prior to model training since machine learning algorithms necessarily require data in numerical form. The content is furthermore processed with Regular Expressions (Regex) to ensure that such email and URL, web addresses, phone numbers, and data points are all consistent, encode symbols, eliminate punctuation and line spacing, and lastly simply convert all text to lowercase, as depicts in Fig 2.12 Throughout this circumstance, Regular Expressions (regex) have been the most approachable technique, which consisted mainly of a compendium of pattern recognition guidance that used acknowledge string sequences in substantial numbers of text input. Those instructions are intended to replicate a text family (alphanumeric, numbers, words), making them adaptable to any text/string class. In essence, regular expressions permit developers to obtain more information from text data while using shorter code.

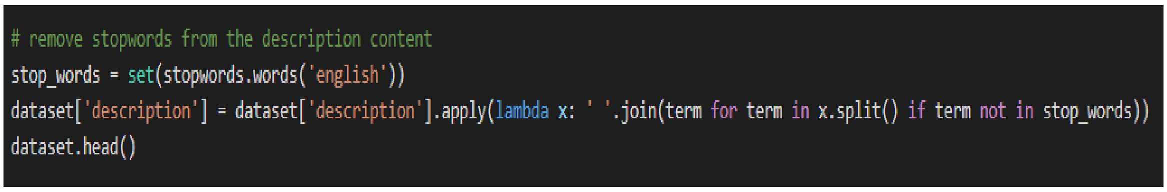


```python
# Replace email address with 'emailaddress'
dataset['description'] = dataset['description'].str.replace(r'^.+@[^\.].*\.[a-z]{2,}$', 'emailaddress')

# Replace urls with 'webaddress'
dataset['description'] = dataset['description'].str.replace(r'^http\://[a-zA-Z0-9\-\.]+\.[a-zA-Z]{2,3}(/\S*)?$',
                                                            'webaddress')

# Replace money symbol with 'money-symbol'
dataset['description'] = dataset['description'].str.replace(r'£|\$', 'money-symbol')

# Replace 10 digit phone number with 'phone-number'
dataset['description'] = dataset['description'].str.replace(r'^\(?[\d]{3}\)?[\s-]?[\d]{3}[\s-]?[\d]{4}$',
                                                            'phone-number')

# Replace normal number with 'number'
dataset['description'] = dataset['description'].str.replace(r'\d+(\.\d+)?', 'number')

# remove punctuation
dataset['description'] = dataset['description'].str.replace(r'[^\w\d\s]', ' ')

# remove whitespace between terms with single space
dataset['description'] = dataset['description'].str.replace(r'\s+', ' ')

# remove leading and trailing whitespace
dataset['description'] = dataset['description'].str.replace(r'^\s+|\s*?$', ' ')

# change words to lower case
dataset['description'] = dataset['description'].str.lower()
```

Figure 2.13 Content proceed with Regular Expressions (Regex)

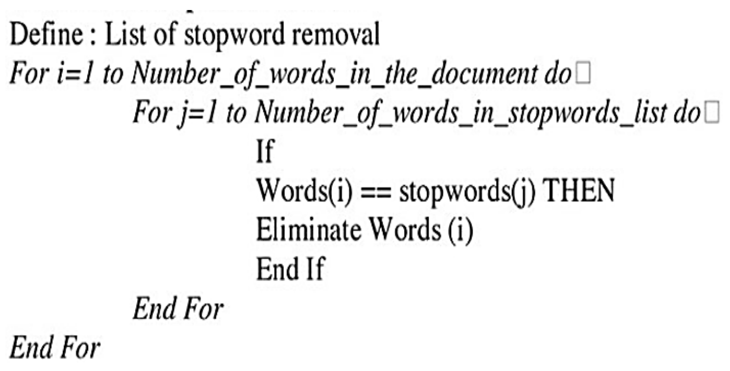
- **Removing stop words with NLTK**

As the succeeding stage of preprocessing the stop words from the text content corpus should be taken out. In natural language processing, futile words (information), are alluded to as stop words. Stop words are terms that have been configured to be disregard by search engines, mostly when pre-defined for searching and extracting them as a consequence of a user's query such as “the”, “a”, “an”, “in”, "but", "because" etc. and so on As displayed underneath Fig 2.13, able to be handled substance more effective by eliminating words that not even operate to any future tasks. Pedcode implementation regarding stop words removal has shown in Fig 2.14



```
# remove stopwords from the description content
stop_words = set(stopwords.words('english'))
dataset['description'] = dataset['description'].apply(lambda x: ' '.join(term for term in x.split() if term not in stop_words))
dataset.head()
```

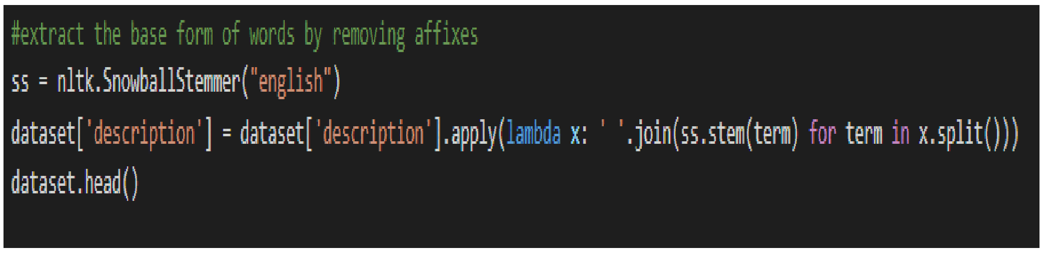
Figure 2.14 Removing the stop words implementation



```
Define : List of stopword removal
For i=1 to Number_of_words_in_the_document do□
        For j=1 to Number_of_words_in_stopwords_list do□
                If
                Words(i) == stopwords(j) THEN
                Eliminate Words (i)
                End If
        End For
End For
```

Figure 2.15 pseudocode stop words removal

- **Stemming with Snowball stemmer**



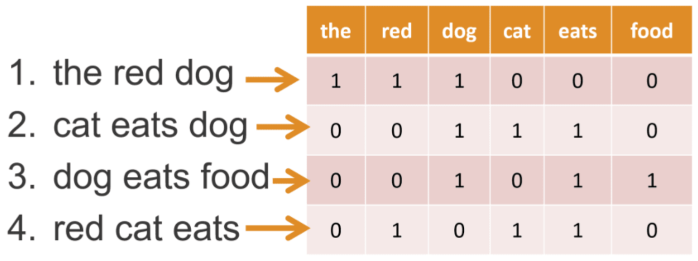
```
#extract the base form of words by removing affixes
ss = nltk.SnowballStemmer("english")
dataset['description'] = dataset['description'].apply(lambda x: ' '.join(ss.stem(term) for term in x.split()))
dataset.head()
```

Figure 2.16 Snowball stemmer inclusion for affixes removal

The base form of words would next be obtained by stripping prefixes and suffixes from each. This is referred to as stems, and it has been shown by cutting down the trees the branches of a tree to its stems. There are various stemming algorithms available, including Porter's Stemmer algorithm, Snowball Stemmer, Lovins Stemmer, Xerox Stemmer, Dawson Stemmer, Krovetz Stemmer, N-Gram Stemmer, Lancaster Stemmer.

A few of these stemming strategies are dynamic and aggressive. Some might be applicable to languages other than English, and the amount of text data impacts various efficiency and speed. As of its computational performance, the Snowball Stemmer was employed as shown in Fig 2.16 for this study while keeping track of Taking control not to over-or under-stem while using these stemming methods.

- **Creating a bag of words BOW**



	the	red	dog	cat	eats	food
1. the red dog	1	1	1	0	0	0
2. cat eats dog	0	0	1	1	1	0
3. dog eats food	0	0	1	0	1	1
4. red cat eats	0	1	0	1	1	0

Figure 2.17 Creating bag of words

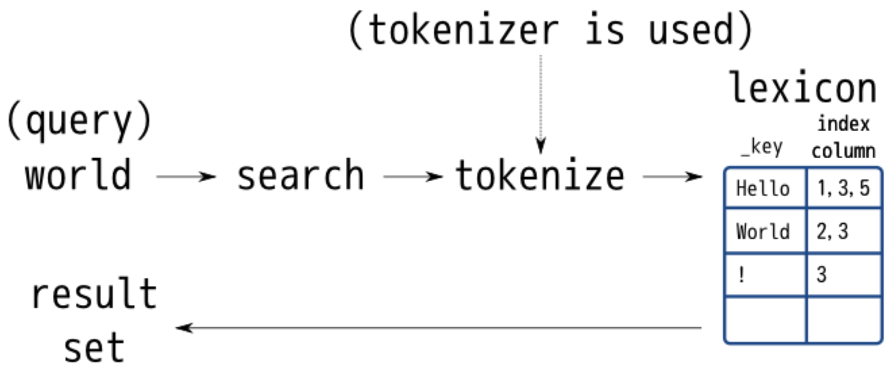
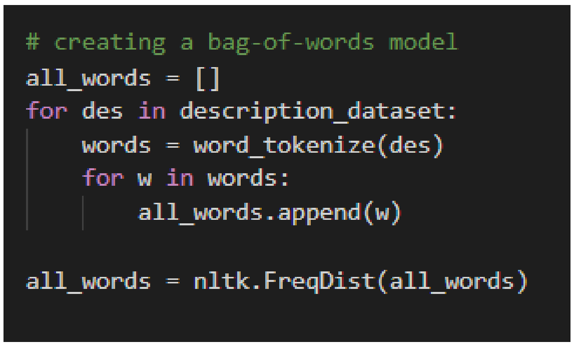


Figure 2.18 Process of BOW

The text content is a concern with modeling since it is unpredictable, and approaches as machine learning algorithms preferred well-defined fixed-length inputs and outputs since ML algorithms cannot operate properly with raw text. The words should always be converted into numbers—specifically, vectors of numbers. The above procedure is referred to as feature extraction or feature encoding.



```
# creating a bag-of-words model
all_words = []
for des in description_dataset:
    words = word_tokenize(des)
    for w in words:
        all_words.append(w)

all_words = nltk.FreqDist(all_words)
```

Figure 2.19 Creating BOW implementation

The Bag of Words (BOW) model of texts is a popular and straightforward simple yet efficient approach to extracting features from text data. Let us break down the contents (text data in sentences) into words. This is a prerequisite in language processing applications where each word must be caught and evaluated. To commence, we construct a BOW model to retrieve characteristics from the text, as depicts in Fig 2.18 And as a result, the top 10 common words in the text data has been plotting as illustrated in Fig 2.19

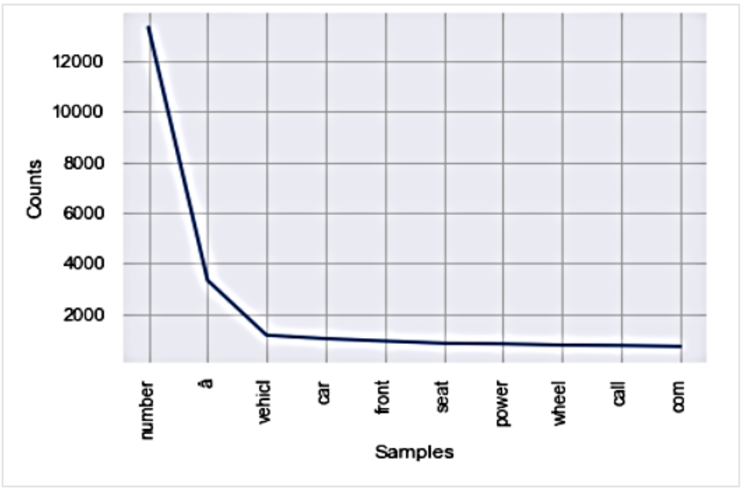


Figure 2.20 Top 10 common words in a corpus

- **NLP technique—term frequency-inverse document frequency**

Next, one concern with assessing word frequency is that extremely common words tend to dominate the content (e.g., higher score), but it might not convey substantially much “informative content” to the models as scarce but perhaps domain-specific keywords.

One method is to dynamically adjust the term frequency relying on how consistently they appear in all documents, such that prominent terms like "the" that is particularly frequent throughout all texts are penalized. As a conclusion, to determine how prominent terms are in the textual information and convert the gathered characters into numerical arrays. The easiest and most effective way of accomplishing this is to employ counts or percentages of each character in the document (known as term frequency).

This is a straightforward method known as a count vectorizer. However, this will excessively exaggerate characters (elevated values) that would have been included in documents of all classes. In an essence, this approach includes what such a "relevant term" is. As a compromise, employed the inverse document frequencies to standardize term frequency values. This approach might assign more weight to tokens with minimal document frequency. Thus, The Term Frequency - Inverse Document Frequency (TF-IDF) vectorization method is utilized in this approach. This NLP technique's TF-IDF model would be collected and stored (serialized) to the local disk for subsequent utilization when converting the training dataset for the software application later.

$$TF\text{-}IDF(t; d) = TF(t; d) : IDF(t;D)$$

It is worth noted that once the term appears in all documents, it has an IDF of 0, whereas when it appears in less than a small percentage of documents, it has a high IDF. As a result, IDF qualifies (multiplies) the value indicated by the word frequency. Some examples include words as "buy," "sell," and so on may occur in any posts, despite category. Consequently, they will offer relatively less to the analysis Implementing IDF will assist us in lowering the value of such tokens. As an aspect of this effort, I employed the TF-IDF technique. At the culmination of all of this, I would be capable of converting the set of descriptions acquired from the dataset into a vector of numbers, which could then be fed into a classifier algorithm. All the procedures described in this preprocessing part were carried out with the aid of Scikit Learn, a Python machine learning package.

- **Data splitting**

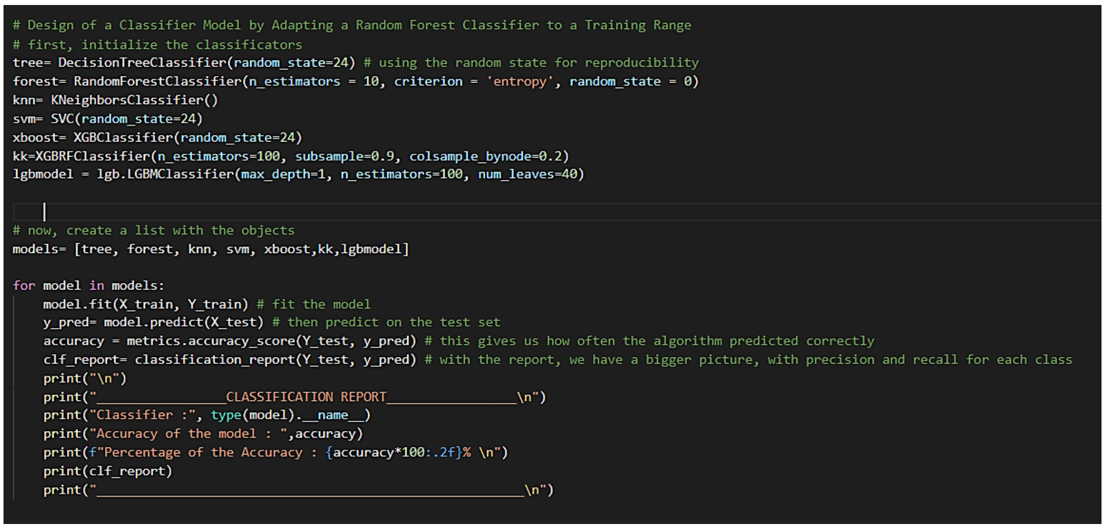
The resultant data frame has the dimensions 5572 by 6506. To train and validate the performance of our classification model, we must partition the dataset into training and testing datasets. Eventually, the training set should indeed be partitioned into train and validation sets. The training dataset is the set of data that was utilized to construct the model which contains real and genuine datasets on which we evaluate the model, and this data can be observed and processed by the model. The validation data has been used to assess a particular model, albeit on a regular basis. This dataset is being used by machine learning experts to fine-tune the model's hyperparameters. As a result, the model encounters this data infrequently and yet never “realizes” from it. The validation set outputs are used to upgrade advanced-level hyperparameters. Consequently, the validation set has an impact on a model, and yet only indirectly. The validation set is sometimes referred to as the Development set or Dev set. This makes perfect sense because this data is beneficial during the model's "advancement" stage. Each classifier was trained and tested in the same 75:25 ratio until the optimal model would be exposed.

3.3.3 Train classifiers on data

- **Classifier selection**

With the advent of web digital innovations, customers already have the potential to peruse, compose, and convey their sales on various classifieds platforms online. With these immense amounts of data, determining a pivotal purpose and the legitimacy that concerns individuals would be especially significant currently. Since the fraudulent activities may occur and most of them not detect prior to posting so that it caused to decrease in the reputation of the sites.

This data is enormous and expressed in natural languages in an inconsistent unstructured textual format. We could not assess or undertake any procedures on this raw and unstructured information unless we try to organize it. And to accomplish so, employed classification through supervised machine learning. I was capable to analyze the dataset versus several of the highly recommended supervised learning algorithms after upgrading it into data preparation, data analysis. After obtaining the training and testing, the data from the train set is vectorized, and indeed the vectorizing model is also stored. All the algorithms we evaluated were employed to develop distinct models by using vectorized train data.



```
# Design of a Classifier Model by Adapting a Random Forest Classifier to a Training Range
# first, initialize the classificators
tree= DecisionTreeClassifier(random_state=24) # using the random state for reproducibility
forest= RandomForestClassifier(n_estimators = 10, criterion = 'entropy', random_state = 0)
knn= KNeighborsClassifier()
svm= SVC(random_state=24)
xboost= XGBClassifier(random_state=24)
kk=XGBRFClassifier(n_estimators=100, subsample=0.9, colsample_bynode=0.2)
lgbmodel = lgb.LGBMClassifier(max_depth=1, n_estimators=100, num_leaves=40)


# now, create a list with the objects
models= [tree, forest, knn, svm, xboost,kk,lgbmodel]

for model in models:
    model.fit(X_train, Y_train) # fit the model
    y_pred= model.predict(X_test) # then predict on the test set
    accuracy = metrics.accuracy_score(Y_test, y_pred) # this gives us how often the algorithm predicted correctly
    clf_report= classification_report(Y_test, y_pred) # with the report, we have a bigger picture, with precision and recall for each class
    print("\n")
    print("_________________CLASSIFICATION REPORT_________________\n")
    print("Classifier :", type(model).__name__)
    print("Accuracy of the model : ",accuracy)
    print(f"Percentage of the Accuracy : {accuracy*100:.2f}% \n")
    print(clf_report)
    print("_______________________________________________________\n")
```

Figure 2.21 Implementation of classifier selection

Above Fig 2.15 showed the implementation of classifier selection. In this study, we examine numerous well-recognized Machine Learning Algorithms that were all implemented using the Scikit Learn machine learning toolkit. These are as follows :

- Decision Tree classifier
- Random Forest classifier
- Support Vector Machines classifier
- KNN classifier
- XGBoost classifier
- XGBRF classifier
- Light Gradient Boosting Machine (LightGBM) classifier

- **Optimum model selection**

  LightGBM is partitioned the tree leaf-by-leaf, as a contrast to existing boosting algorithms, which extend the tree level-by-level. It selects the leaf with the highest delta reduction to thrive.

  The leaf-wise approach has a narrower loss than the level-wise technique since the leaf is consistent. In sparse datasets, leaf-wise tree expansion may enhance the model's sophistication and leads to imbalanced datasets which are referred to as overfitting. The following illustration depicts Leaf-Wise Tree Growth:

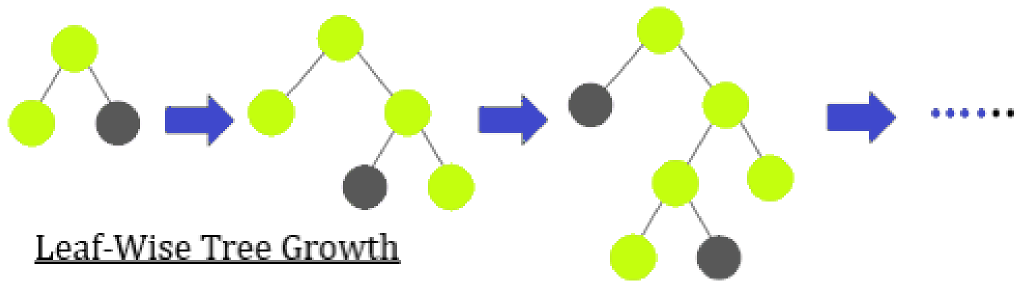


Figure 2.22 LightGBM leaf-wise growth

This boosting algorithm comprises a methodology for gradient boosting that employs a tree-based learning approach. It has the following benefits:

➢ **Increased training efficiency and higher speed**: Light GBM employs a histogram-based approach. It divides continuous selected features into discrete bits, which accelerates up the training processes.

➢ **Decreased memory consumption**: Substitutes continuous data with discrete bits, allowing in fewer computational requirements.

➢ **Improved precision than other boosting algorithms**: Its conclusions and recommendations more sophisticated trees by implementing a leaf-wise split procedure rather than a level-wise split procedure, which is the major contributor in obtaining increased precision. Nonetheless, it can potentially cause overfitting, which could be mitigated by adjusting the max depth option.

- **The capability of processing enormous amounts of data**: When contrasted to XGBOOST, it is the potential of performing equivalent well enough with massive datasets while consuming significantly reduced training time.

- **Concurrent and GPU learning are supported.**

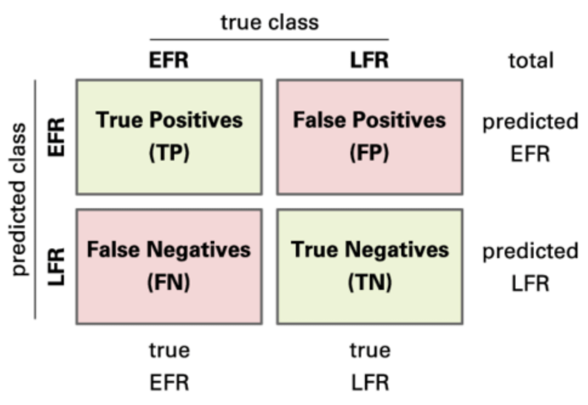
  Each of the classifiers was implemented and tested until the optimum model in terms of F1 score (accuracy and recall) and confusion matrix was being demonstrated. Taking into consideration all of the aspects indicated above, as well as the analysis and evaluation segment, which is briefly discussed later, the classification matrix indisputably revealed LightGBM as the highest effective model to pursue with further implementations.

3.3.1 Analysis & Evaluation

Models might indeed be assessed considering a variety of metrics. Nevertheless, establishing the appropriate assessment measure is essential and typically relies on the problem being addressed. A comprehensive grasp of a wide range of metrics can facilitate the assessor in determining an entirely acceptable correlation between the problem description and a metric.

- Confusion matrix

  The confusion matrix is a prominent metric for addressing classification concerns. It can be used for both binary classification and multi-class classification situations. It effectively assists you in graphically evaluating how the model is functioning properly. The matrix may indeed be constructed for each classification model prediction to indicate the number of test cases accurately and probably wrongly categorized. It appears such as this (presuming that the targeted classes are 1 - Positive and 0 - Negative). A proper visualization confusion matrix is as follows depicted in Fig 2.17



$$\mathrm{PR} = \frac{\mathrm{TP}}{\mathrm{TP+FP}}$$
$$\mathrm{RE} = \frac{\mathrm{TP}}{\mathrm{TP+FN}}$$
$$\mathrm{CA} = \frac{\mathrm{TP+TN}}{\mathrm{TP+TN+FP+FN}}$$
$$\mathrm{F}_1 = \frac{\mathrm{2TP}}{\mathrm{2TP+FP+FN}}$$

Figure 2.23 Confusion matrix

TN: Number of clearly and precisely assessed negative instances

TP: The number of clearly and precisely assessed positive instances.

FN: The number of positive instances which have been erroneously categorized as negative.

FP: The number of negative instances which have been clearly and accurately categorized as positive.

- Accuracy

The most basic measurement is accuracy, which is calculated as the amount of properly categorized test instances divided by the total number of test cases. And can be used to tackle most common concerns, although it is relatively ineffective when interacting with imbalanced datasets. For instance, if we detect fraud in records, the ratio of fraud to non-fraud instances might be as high as 1:99. If accuracy is employed in such instances, the model will be 99 percent accurate and reliable by forecasting all test cases as non-fraud. The model that is 99 percent accurate will be profoundly pointless. If a model is inadequately trained, it will miss out on the 10 fraudulent data points if it forecasts all 1000 (says) sample values as non-frauds. If the model's accuracy is assessed, it will indicate that it successfully predicted 990 points, yielding it an accuracy of (990/1000)*100 = 99 %! As a result, accuracy is a deceptive measure of the model's integrity. Therefore, in such a circumstance, a measurement that can focus down on the 10 fraud data points that the model entirely missed is required.

- F1 Score

  The F1 score is the coefficient of determination of Recall and Precision and therefore balanced out the core strengths of both; moreover, this score encompasses both false negatives and false positives. It is not as counterintuitive as accuracy, but F1 is typically far more beneficial than accuracy, specifically if the classification is unequally distributed. Accuracy performs effectively when the cost of false positives and false negatives is fairly comparable. If the cost of false positives and false negatives varies significantly, it is advantageous to involve both Precision and Recall. In our scenario, the F1 score shows in the results section.

F1 Score = 2((Recall * Precision) / (Recall + Precision))*

Considering all the used classification matrices mentioned above, conclude the performance matrix for the selected model is the F1 score. Thus, the results were obtained using the classification matrix depicted in the results section.

3.3.2 Hyperparameter tweaking

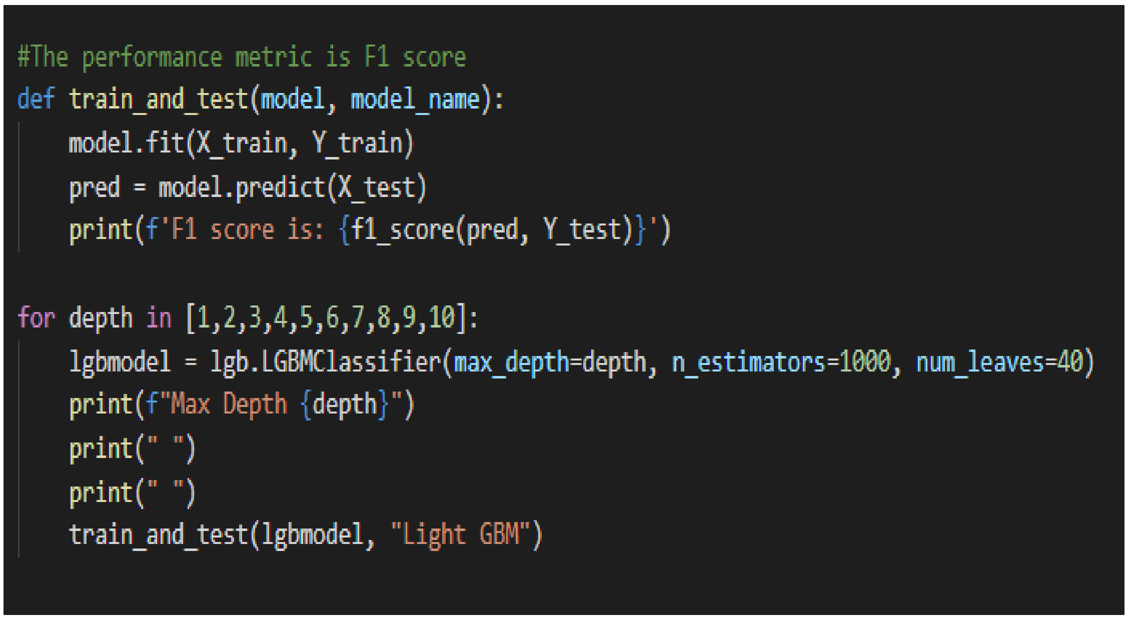
LightGBM is already one of the foremost prevalently exploited machine learning methods. It has been well for its propensity to win ML challenges. Many publications commend it and emphasize its strengths over alternative approaches, considering it must-have expertise for machine learning enthusiasts. When building a machine learning model, we should be allotted design alternatives for specifying getting a fine model architecture.

We do not always know what the best model architecture is for a selected model, hence we should experiment with a variety of options. Inauthentic machine learning sort of manner, we'll presumably enquire the machine to undertake these investigations and choose the optimized model architecture instantaneously. The parameters that define the model are acknowledged as hyperparameters, and the procedures of finding the greatest model architecture are known as hyperparameter tuning.

Although LightGBM is quite quick and efficient, running a script on a typical laptop may be complicated: when fitting a machine learning model, it generally includes hyperparameter tweaking and — though not always — cross-validation. If all permutations of model parameters are explored, tweaking may exponentially augment the computational and storage consumption.

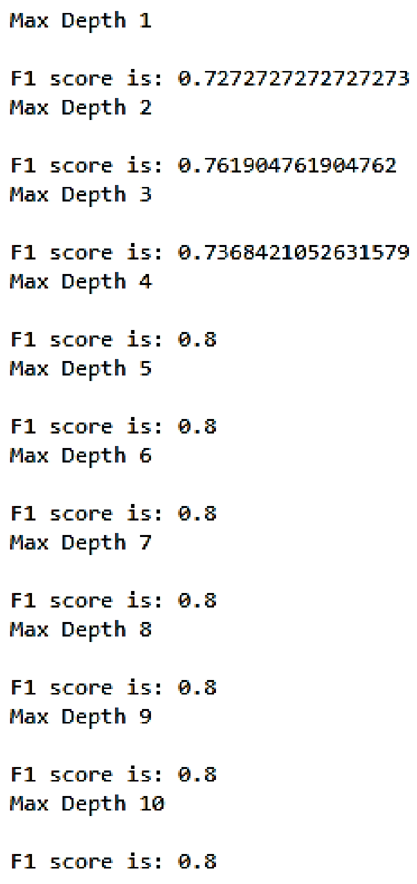
As a result, an extensive investigation for optimal parameters could be out of the discussion. This section discussed approaches for LightGBM hyperparameter tweaking to strike a fair balance between model performance and computational resource constraints.

The F1 score is the project's performance indicator as shown in the implementation of the performance matrix in Fig 2.23 To construct the score, this metric takes precision and recall into considerations. The F1 score comprises the highest value of 1 and the lowest value of 0.



```
#The performance metric is F1 score
def train_and_test(model, model_name):
    model.fit(X_train, Y_train)
    pred = model.predict(X_test)
    print(f'F1 score is: {f1_score(pred, Y_test)}')

for depth in [1,2,3,4,5,6,7,8,9,10]:
    lgbmodel = lgb.LGBMClassifier(max_depth=depth, n_estimators=1000, num_leaves=40)
    print(f"Max Depth {depth}")
    print(" ")
    print(" ")
    train_and_test(lgbmodel, "Light GBM")
```

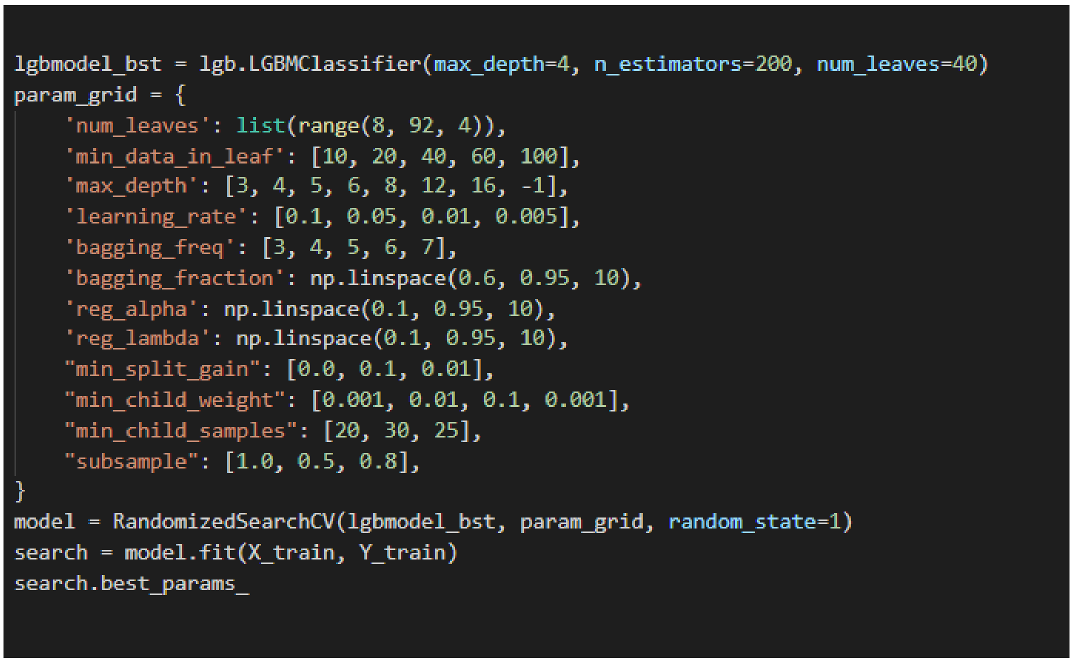
Figure 2.24 Highest Max_depth obtained from F1 score metric implementation



```
Max Depth 1

F1 score is: 0.7272727272727273
Max Depth 2

F1 score is: 0.761904761904762
Max Depth 3

F1 score is: 0.7368421052631579
Max Depth 4

F1 score is: 0.8
Max Depth 5

F1 score is: 0.8
Max Depth 6

F1 score is: 0.8
Max Depth 7

F1 score is: 0.8
Max Depth 8

F1 score is: 0.8
Max Depth 9

F1 score is: 0.8
Max Depth 10

F1 score is: 0.8
```

Figure 2.25 Results obtained after the Max_depth iteration

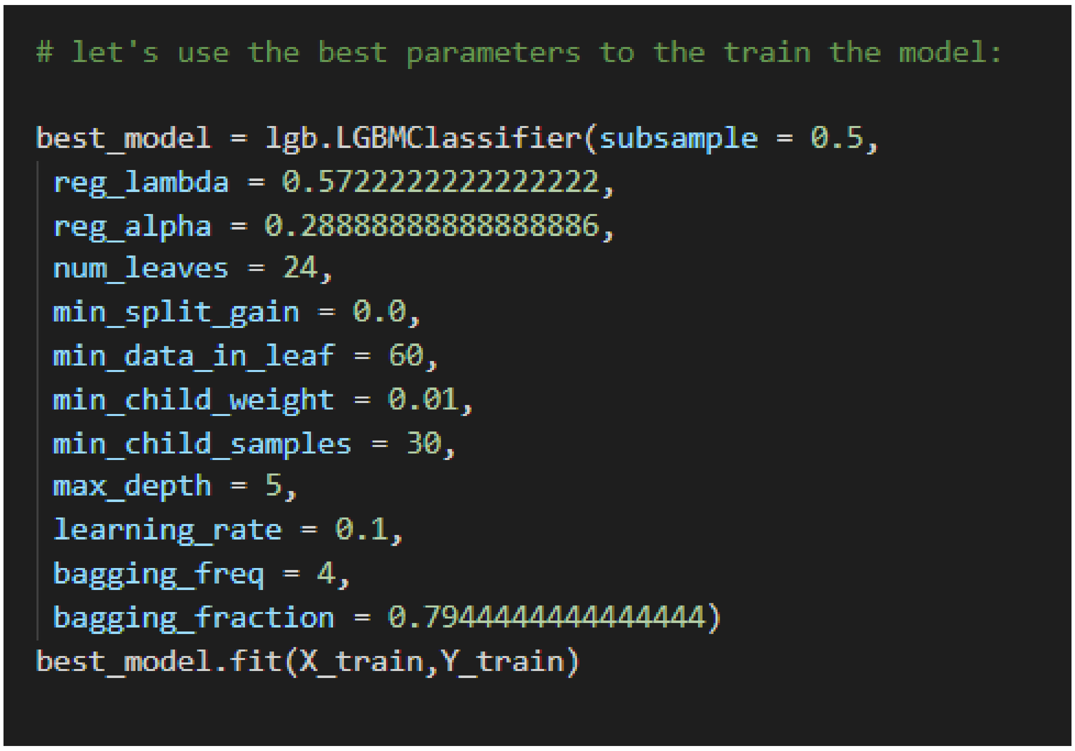
The random forest learning technique in Python contains the well-known scikit-learn function RandomizedSearchCV, which is selected only a few samples. LightGBM, an ensemble learning architecture, can often outperform a random forest model that well-tuned. Instead of offering a discrete set of values to examine for each hyperparameter, RandomizedSearchCV aims to provide a comprehensive dispersion or enumeration of hyperparameters. Values for the numerous hyperparameters are drawn at random from this range. RandomizedSearchCV Python implementation for the Random Forest method is shown below. Nevertheless, I was able to acquire a randomized SearchCV that was compatible with LightGBM as shown in Fig 2.25



```
lgbmodel_bst = lgb.LGBMClassifier(max_depth=4, n_estimators=200, num_leaves=40)
param_grid = {
    'num_leaves': list(range(8, 92, 4)),
    'min_data_in_leaf': [10, 20, 40, 60, 100],
    'max_depth': [3, 4, 5, 6, 8, 12, 16, -1],
    'learning_rate': [0.1, 0.05, 0.01, 0.005],
    'bagging_freq': [3, 4, 5, 6, 7],
    'bagging_fraction': np.linspace(0.6, 0.95, 10),
    'reg_alpha': np.linspace(0.1, 0.95, 10),
    'reg_lambda': np.linspace(0.1, 0.95, 10),
    "min_split_gain": [0.0, 0.1, 0.01],
    "min_child_weight": [0.001, 0.01, 0.1, 0.001],
    "min_child_samples": [20, 30, 25],
    "subsample": [1.0, 0.5, 0.8],
}
model = RandomizedSearchCV(lgbmodel_bst, param_grid, random_state=1)
search = model.fit(X_train, Y_train)
search.best_params_
```

Figure 2.26 RandomizedGridSearchCV for LightGBM

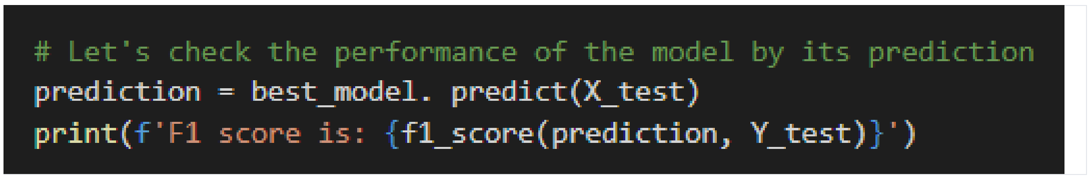
After implemented the above Random grid search CV, we obtained the parameters best values as follows :



```python
# let's use the best parameters to the train the model:

best_model = lgb.LGBMClassifier(subsample = 0.5,
 reg_lambda = 0.5722222222222222,
 reg_alpha = 0.28888888888888886,
 num_leaves = 24,
 min_split_gain = 0.0,
 min_data_in_leaf = 60,
 min_child_weight = 0.01,
 min_child_samples = 30,
 max_depth = 5,
 learning_rate = 0.1,
 bagging_freq = 4,
 bagging_fraction = 0.7944444444444444)
best_model.fit(X_train,Y_train)
```

Figure 2.27 Obtained hyperparameter values for LightGBM

After we utilized the above Fig 2.26 depicted optimum hyperparameter values to train the LightGBM classifier respectively. Once the training was accomplished with optimum values, its prediction was scrutinized by evaluating the performance of the model.



```python
# Let's check the performance of the model by its prediction
prediction = best_model. predict(X_test)
print(f'F1 score is: {f1_score(prediction, Y_test)}')
```

Figure 2.28 Performance review

As the very last step of this phase as shown in Fig 2.27, we performed extensive training on the dataset as then our web application can make predictions on content that has not previously been encountered. Ultimately, the model was serialized using pickle.

3.3.3 Fraud detection server implementation

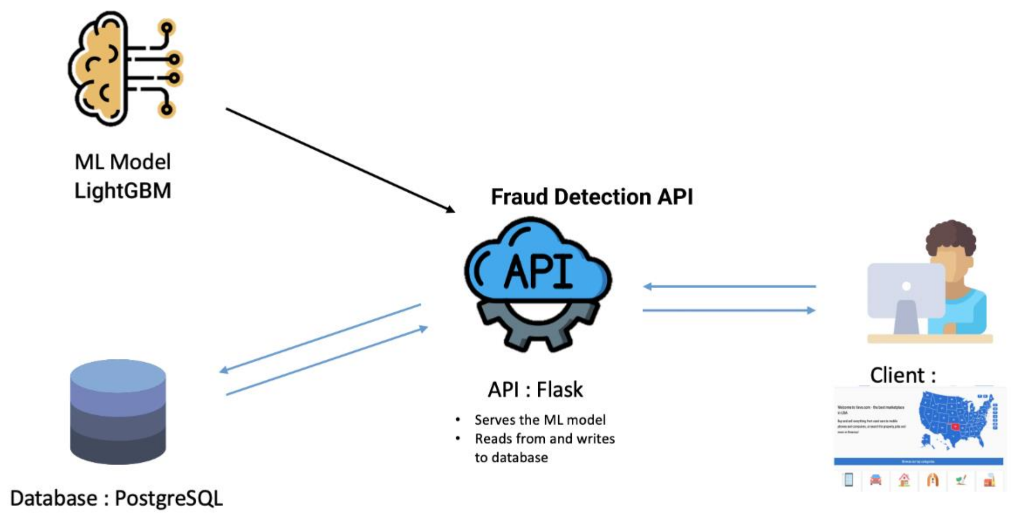


Figure 2.29 Client-Server design of fraud detection system

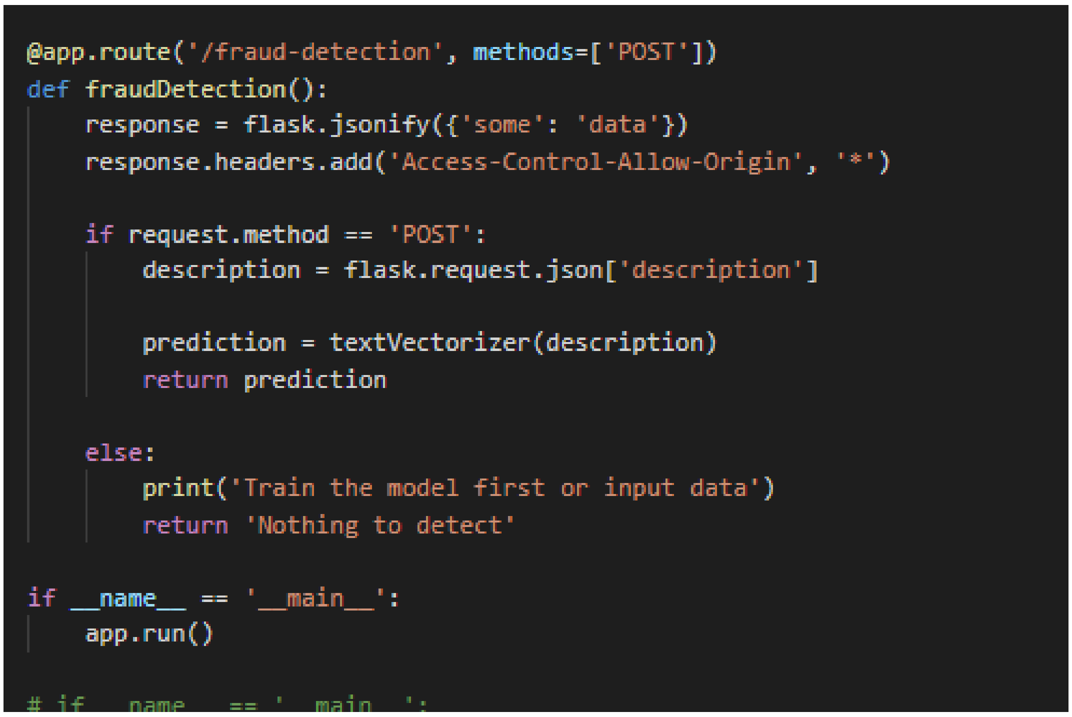
Model initial deployment is one of the key stages of any machine learning endeavor, and it may be onerous. In the end, The model should be offered to end customers so that they could utilize it. How would our machine learning model be offered to its user? Subsequently, when questions arise as what are the many concerns that must address while placing the model into production? And how do you begin implementing a model? Flask now simply initiates the task.

What exactly is Flask?

Flask is a Python-based development platform, and it encompasses a series of modules that allow a web developer to build programs without needing to bother about complexities such as the set of protocols management, threading management, and so on. Flask equips us with a range of possibilities for constructing web applications, and even the required tools and libraries to establish a web application.

Now that having the trained model, we were able to create a Flask application thatreads input descriptions prior to posting via the web application. First, build an instance of the Flask class which will accept the current module name,__name__ as a parameter.

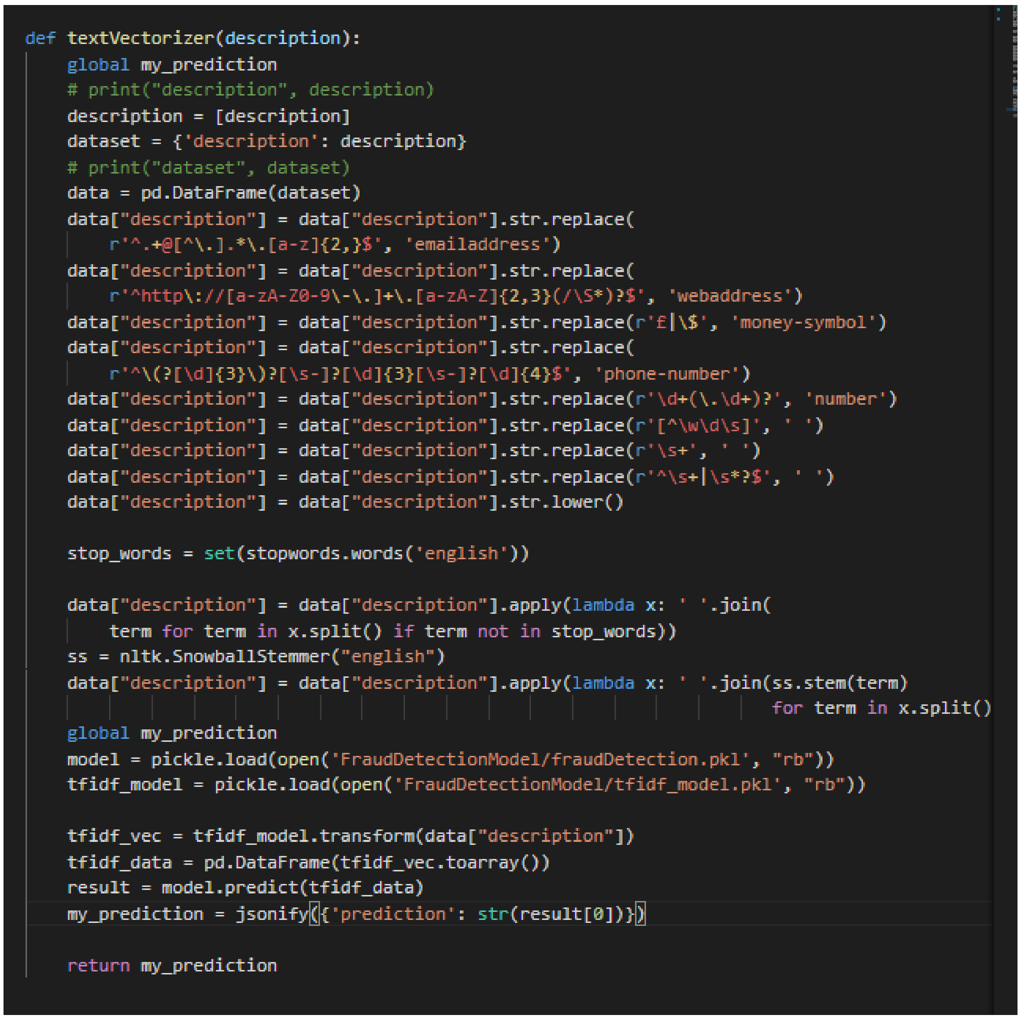
Now, whenever a customer submits a short description, the flask API detects a post request and invokes the fraudDetection() method as shown in Fig 2.29, where we retrieve the form data with description and afterward redirect to the textVectorizer() method as depicted in Fig 2.30, where we implement all the preceding preprocessing approaches. The model was then embedded into the same procedure.



```python
@app.route('/fraud-detection', methods=['POST'])
def fraudDetection():
    response = flask.jsonify({'some': 'data'})
    response.headers.add('Access-Control-Allow-Origin', '*')

    if request.method == 'POST':
        description = flask.request.json['description']

        prediction = textVectorizer(description)
        return prediction

    else:
        print('Train the model first or input data')
        return 'Nothing to detect'

if __name__ == '__main__':
    app.run()

# if __name__ == '__main__':
```

Figure 2.30 Flask API fraudDetection() implementation

Finally, utilizing imported model pickles returns the detection output when the preceding implementation has been accomplished.



```python
def textVectorizer(description):
    global my_prediction
    # print("description", description)
    description = [description]
    dataset = {'description': description}
    # print("dataset", dataset)
    data = pd.DataFrame(dataset)
    data["description"] = data["description"].str.replace(
        r'^.+@[^\.].*\.[a-z]{2,}$', 'emailaddress')
    data["description"] = data["description"].str.replace(
        r'^http\://[a-zA-Z0-9\-\.]+\.[a-zA-Z]{2,3}(/\S*)?$', 'webaddress')
    data["description"] = data["description"].str.replace(r'£|\$', 'money-symbol')
    data["description"] = data["description"].str.replace(
        r'^\(?[\d]{3}\)?[\s-]?[\d]{3}[\s-]?[\d]{4}$', 'phone-number')
    data["description"] = data["description"].str.replace(r'\d+(\.\d+)?', 'number')
    data["description"] = data["description"].str.replace(r'[^\w\d\s]', ' ')
    data["description"] = data["description"].str.replace(r'\s+', ' ')
    data["description"] = data["description"].str.replace(r'^\s+|\s*?$', ' ')
    data["description"] = data["description"].str.lower()

    stop_words = set(stopwords.words('english'))

    data["description"] = data["description"].apply(lambda x: ' '.join(
        term for term in x.split() if term not in stop_words))
    ss = nltk.SnowballStemmer("english")
    data["description"] = data["description"].apply(lambda x: ' '.join(ss.stem(term)
                                                                    for term in x.split()
    global my_prediction
    model = pickle.load(open('FraudDetectionModel/fraudDetection.pkl', "rb"))
    tfidf_model = pickle.load(open('FraudDetectionModel/tfidf_model.pkl', "rb"))

    tfidf_vec = tfidf_model.transform(data["description"])
    tfidf_data = pd.DataFrame(tfidf_vec.toarray())
    result = model.predict(tfidf_data)
    my_prediction = jsonify({'prediction': str(result[0])})

    return my_prediction
```

Figure 2.31 Flask API textVectorizer() implementation

A conclusion of implementation we can roughly summarize as the model was serialized using Pickle and lodged with the Flask REST API's 'fraud-detection endpoint for efficient fraud detection as classifying them into 'spam' or 'ham' and visualization through the advertisement submission form in 'Tievs' front-end app.

3.3.4 User interface implementation

The user interface is essential for satisfying user requirements and ensuring the smooth operation of our website. Through contrasting aesthetics, clean design, and responsiveness, a very well user interface allows more efficient communication between the client and the program or app. Front-end implementation was done mainly focusing on user interaction with an easy and straightforward experience. User has to log in and post their ads by filling the advertisements form the description input where it will assist as the main feature of detect inappropriate/fraudulent content by utilizing the cutting-edge technologies. Consequently, the suggested solution's goal of outperforming previous classified advertising systems in providing customers' needs by employing the most profitable, time-saving, human-centric, and error-preventive techniques was met by having these appealing user interfaces.

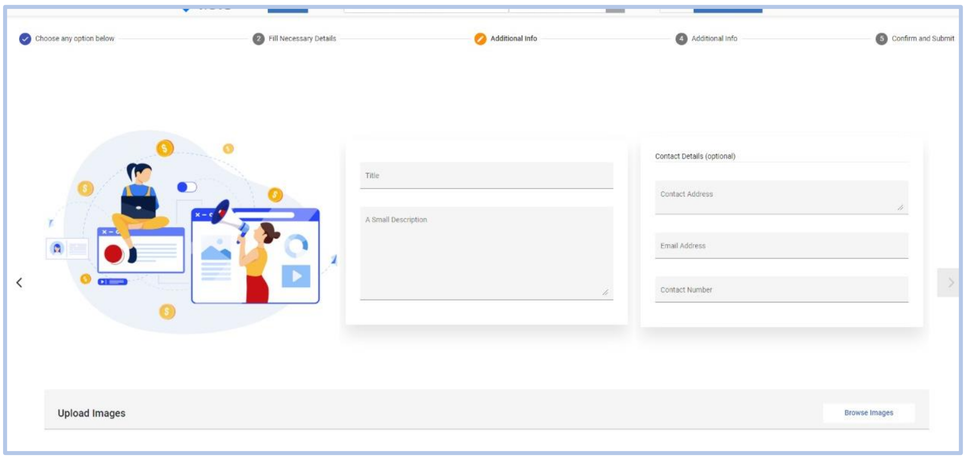


Figure 2.32 User interface of the ad submission form

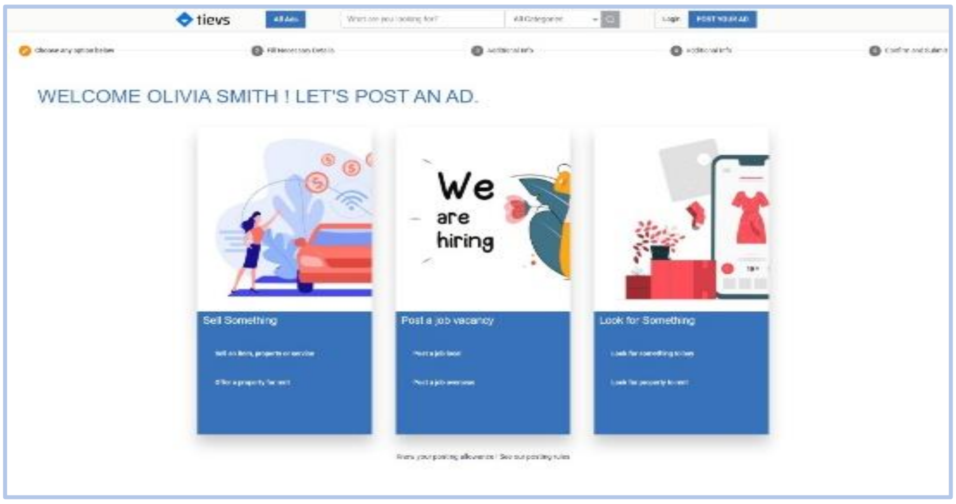


Figure 2.33 User interface of the 'TIEVS

3.3.5 Testing

Testing was conducted to ensure that the model functionalities performed as anticipated, as well as to ascertain the algorithm's accuracy and reliability, computational efficiency. Model precision, and recall, and perhaps to exhibit the ROC curve and Confusion Matrix were analyzed in developer mode. Clearly showed the validated postman results and retrain employing two data sets with varied records to effectively analyze the model's performance in production mode. Using the previously mentioned classification matrix and test case, the main initiative was to recognize the most effective model and have the most out of the divulged model. Tables 1, 2,3 showed the test cases accomplished throughout the research study. The important results/observations of the production mode testing are addressed in the chapter below titled "Results and Discussion."

Table 2.1 Test Case 01

ID	**01**
Test Case	Developer mode testing for evaluating the performance of selected algorithms – **Support Vector Machine**
Input	‘Car_dataset2.csv' comprises 20000 data records including descriptions & class labels as ‘spam' and ‘ham.'
Expected Output	Classification report including precision level, confusion matrix visualization, and k-fold cross-validation

Table 2.2 Test Case 02

ID	**02**
Test Case	Developer mode testing for evaluating the performance of selected algorithms – **Random Forest Classifier**
Input	‘Car_dataset2.csv' comprises 20000 data records including descriptions & class labels as ‘spam' and ‘ham.'
Expected Output	Classification report including precision level, confusion matrix visualization, and k-fold cross-validation

Table 2.3 Test Case 03

ID	**03**
Test Case	Developer mode testing for evaluating the performance of selected algorithms –**Kneighbors Classifier**
Input	‘Car_dataset2.csv' comprises 20000 data records including descriptions & class labels as ‘spam' and ‘ham.'
Expected Output	Classification report including precision level, confusion matrix visualization, and k-fold cross-validation

Table 2.4 Test Case 04

ID	**04**
Test Case	Developer mode testing for evaluating the performance of selected algorithms – **XGBRF Classifier**
Input	‘Car_dataset2.csv' comprises 20000 data records including descriptions & class labels as ‘spam' and ‘ham.'
Expected Output	Classification report including precision level, confusion matrix visualization, and k-fold cross-validation

Table 2.5 Test Case 05

ID	**05**
Test Case	Developer mode testing for evaluating the performance of selected algorithms – **XGB Classifier**
Input	‘Car_dataset2.csv' comprises 20000 data records including descriptions & class labels as ‘spam' and ‘ham.'
Expected Output	Classification report including precision level, confusion matrix visualization, and k-fold cross-validation

Table 2.6 Test Case 06

ID	**06**
Test Case	Developer mode testing for evaluating the performance of selected algorithms – **LightGBM Classifier**
Input	‘Car_dataset2.csv' comprises 20000 data records including descriptions & class labels as ‘spam' and ‘ham.'
Expected Output	Classification report including precision level, confusion matrix visualization, and k-fold cross-validation

Table 2.7 Test Case 07

ID	**07**
Test Case	Developer mode testing for evaluating the performance of selected algorithms – **Decision Tree Classifier**
Input	‘Car_dataset2.csv' comprises 20000 data records including descriptions & class labels as ‘spam' and ‘ham.'
Expected Output	Classification report including precision level, confusion matrix visualization, and k-fold cross-validation

Table 2.8 Test Case 08

ID	**08**
Test Case	Testing incorporating real-world input from the user in a non-fraudulent situation
Input	‘Car_dataset2.csv' comprises 20000 data records including descriptions & class labels as ‘spam' and ‘ham.'
Expected Output	Successfully predicted postman result visualization as non-fraudulent /ham

Table 2.9 Test Case 09

ID	**09**
Test Case	Testing incorporating real-world input from the user in a fraudulent situation
Input	‘Car dataset2.csv' comprises 20000 data records including descriptions & class labels as ‘spam' and ‘ham.'
Expected Output	Successfully predicted postman result visualization as fraudulent /spam

Table 2.10 Test Case 10

ID	**10**
Test Case	After integration and deploying mode testing incorporating real-world input from the user in a fraudulent situation
Input	‘Car_dataset2.csv’ comprises 20000 data records including descriptions & class labels as ‘spam’ and ‘ham.’
Expected Output	Successfully predicted result visualization as spam on postman console after integration and cloud deploying

Table 2.11 Test Case 11

ID	**11**
Test Case	Production mode testing incorporating real-world input from the user in a fraudulent situation
Input	‘Car_dataset2.csv' comprises 20000 data records including descriptions & class labels as ‘spam' and ‘ham.'
Expected Output	Successfully predicted result fraud warning visualization on web application

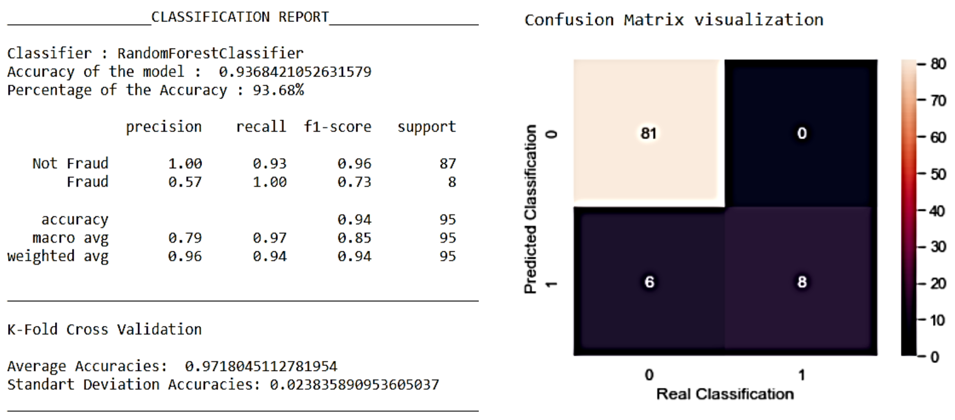
3 RESULTS & DISCUSSION

3.1 Results

The results of both development mode and production mode testing are assessed below. This segment includes screenshots and conclusive evidence of the postman console for better visualization.

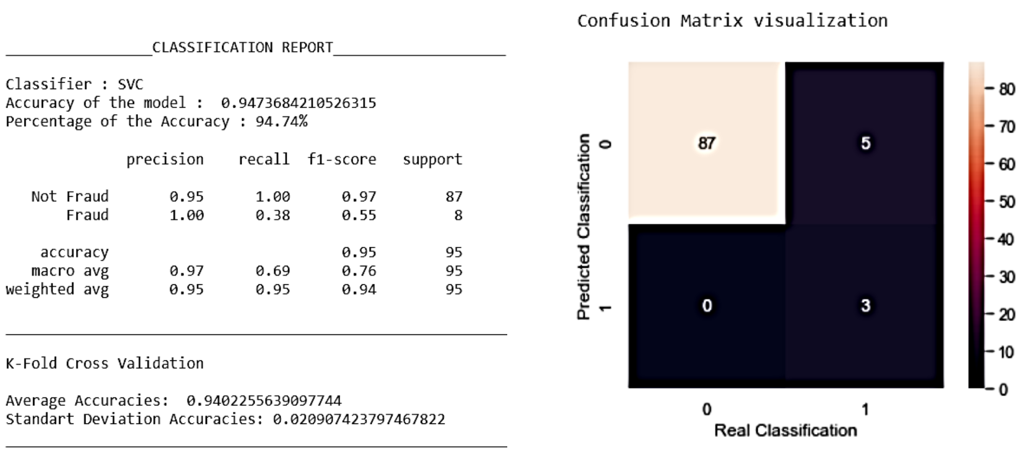
3.1.1 Developer test results

As mentioned test cases 1 and 2 resulting in some of the classification reports along with the confusion matrix and accuracy level are depicted below Fig 3.1 and Fig 32.



```
________________CLASSIFICATION REPORT________________

Classifier : RandomForestClassifier
Accuracy of the model :  0.9368421052631579
Percentage of the Accuracy : 93.68%

              precision    recall  f1-score   support

   Not Fraud       1.00      0.93      0.96        87
       Fraud       0.57      1.00      0.73         8

    accuracy                           0.94        95
   macro avg       0.79      0.97      0.85        95
weighted avg       0.96      0.94      0.94        95

______________________________________________________

K-Fold Cross Validation

Average Accuracies:  0.9718045112781954
Standart Deviation Accuracies: 0.023835890953605037
______________________________________________________
```

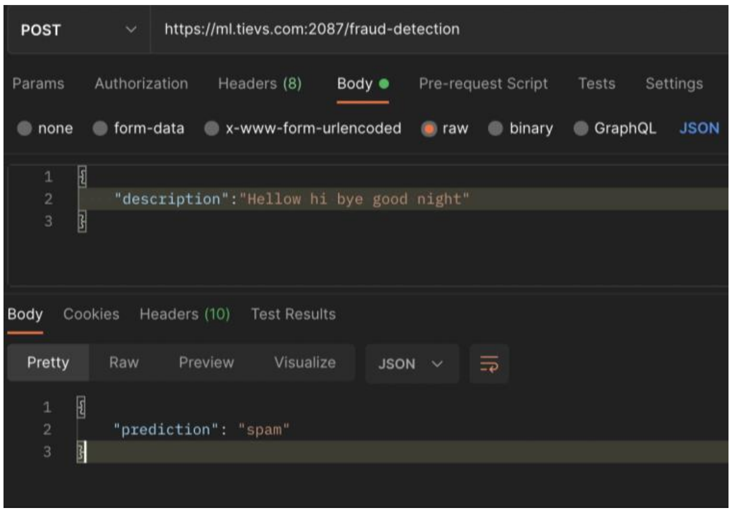
Figure 3.1 Classification report and confusion matrix of RandomForest Classifier



```
_________________CLASSIFICATION REPORT____________________

Classifier : SVC
Accuracy of the model :  0.9473684210526315
Percentage of the Accuracy : 94.74%

              precision    recall  f1-score   support

   Not Fraud       0.95      1.00      0.97        87
       Fraud       1.00      0.38      0.55         8

    accuracy                           0.95        95
   macro avg       0.97      0.69      0.76        95
weighted avg       0.95      0.95      0.94        95

__________________________________________________________

K-Fold Cross Validation

Average Accuracies:  0.9402255639097744
Standart Deviation Accuracies: 0.020907423797467822
__________________________________________________________
```

Figure 3.2 Classification report and confusion matrix of SVC Classifier

3.1.2 Production test results



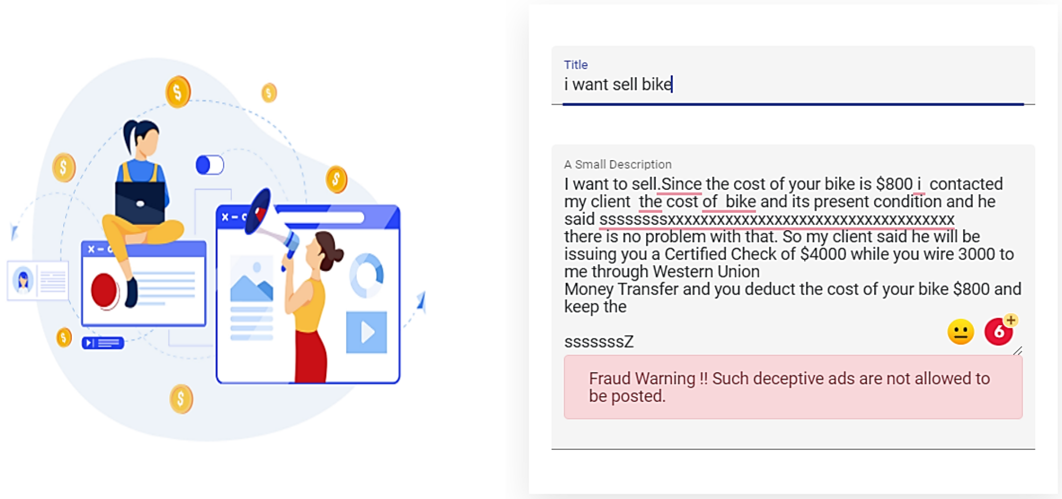
https://ml.tievs.com:2087/fraud-detection

```
{
    "description":"Hellow hi bye good night"
}
```

```
{
    "prediction": "spam"
}
```

Figure 3.3 Postman of fraudulent input - production mode testing result

As mentioned in test case 12, the above Fig 3.3 depicts the postman results in “fraud-detection” end real-world input from the user in a fraudulent situation after integration and could deployment for establishing the production purpose.



i want sell bike

I want to sell.Since the cost of your bike is $800 i contacted my client the cost of bike and its present condition and he said ssssssssxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx there is no problem with that. So my client said he will be issuing you a Certified Check of $4000 while you wire 3000 to me through Western Union
Money Transfer and you deduct the cost of your bike $800 and keep the

sssssssZ

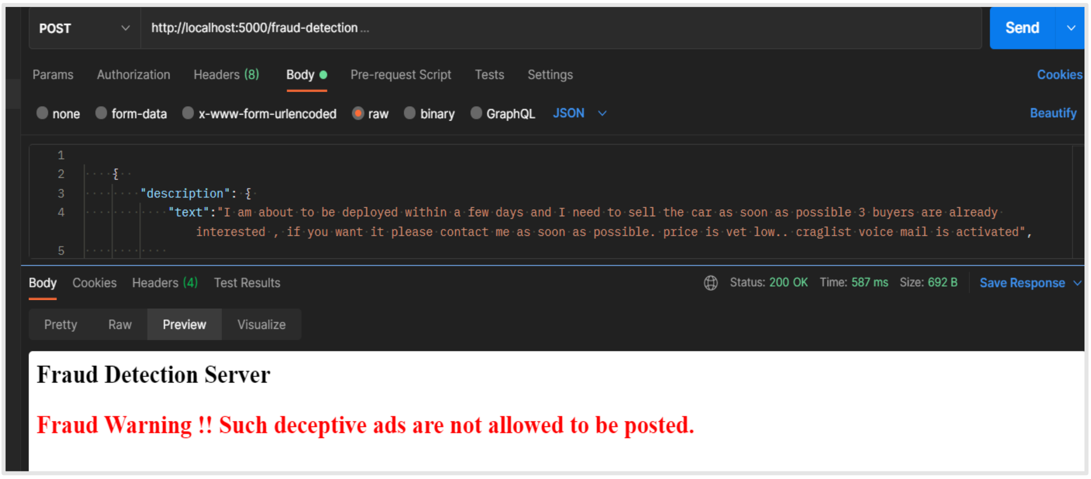
Fraud Warning !! Such deceptive ads are not allowed to be posted.

Figure 3.4 Visualization of fraud warning as the ad posting

As mentioned in test case 12 in the testing section, Successfully predicted result fraud warning visualization was done in the frontend application using actual fraudulent data on the description box for the production purpose shown in the above Fig 3.4

3.1.3 Postman test results

Successfully predicted fraud warning visualization in postman for the local developer testing mode has shown in the below Fig 3.5 regarding the mentioned test cases 12 in the testing section.



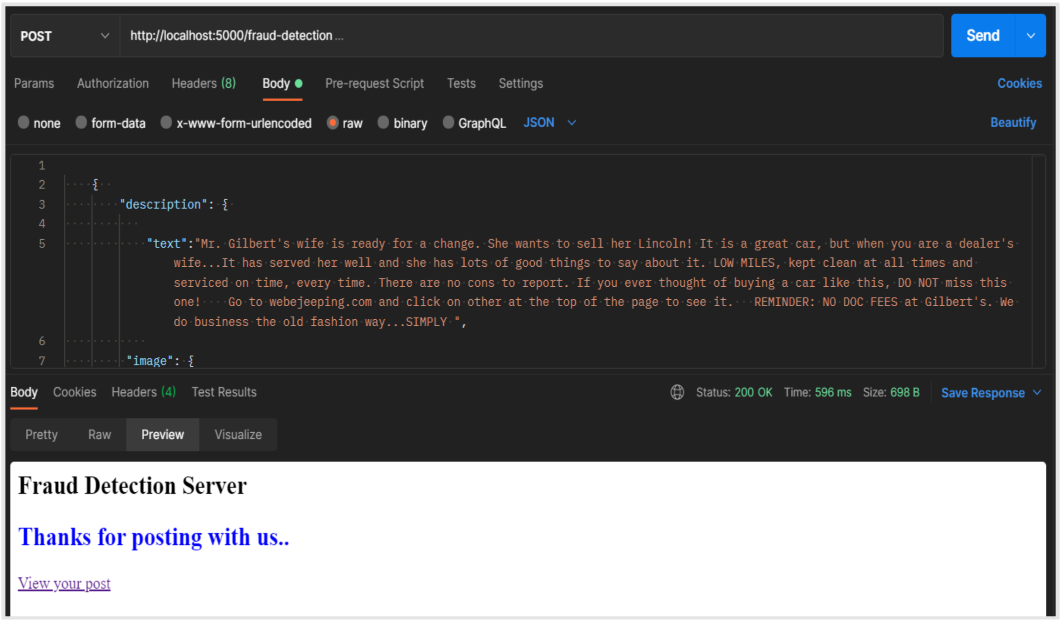
```
{
    "description": {
        "text":"I am about to be deployed within a few days and I need to sell the car as soon as possible 3 buyers are already
            interested , if you want it please contact me as soon as possible. price is vet low.. craglist voice mail is activated",
```

Fraud Detection Server

Fraud Warning !! Such deceptive ads are not allowed to be posted.

Figure 3.5 Postman of fraudulent input testing result

Successfully predicted non-fraud (ham) visualization in postman for the local developer testing mode has shown in the below Fig 3.6 regarding the mentioned test cases 12 in the testing section.



```
{
    "description": {

        "text":"Mr. Gilbert's wife is ready for a change. She wants to sell her Lincoln! It is a great car, but when you are a dealer's
            wife...It has served her well and she has lots of good things to say about it. LOW MILES, kept clean at all times and
            serviced on time, every time. There are no cons to report. If you ever thought of buying a car like this, DO NOT miss this
            one!    Go to webejeeping.com and click on other at the top of the page to see it.   REMINDER: NO DOC FEES at Gilbert's. We
            do business the old fashion way...SIMPLY ",

    "image": {
```

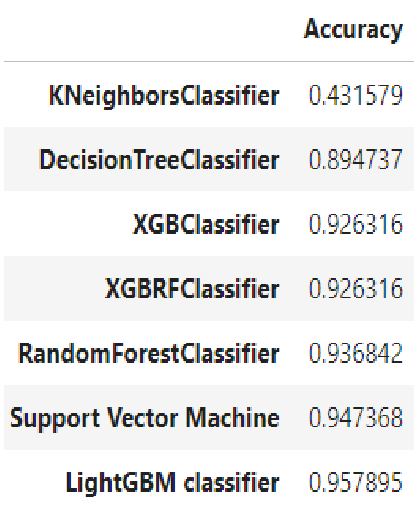
Fraud Detection Server

Thanks for posting with us..

View your post

Figure 3.6 Postman of legitimate input testing result

3.1.4 Optimum model results



	Accuracy
KNeighborsClassifier	0.431579
DecisionTreeClassifier	0.894737
XGBClassifier	0.926316
XGBRFClassifier	0.926316
RandomForestClassifier	0.936842
Support Vector Machine	0.947368
LightGBM classifier	0.957895

Figure 3.7 Accuracy levels of classifiers

Accuracy level has been obtained as shown in Fig 3.7 to have an optimum model after successfully compared between seven algorithms. And also lastly decided best model was confirmed as LightGBM classifier according to the classification report. Below Fig 3.1.4.2. has depicted the confusion matrix obtained after parameter tweaking.

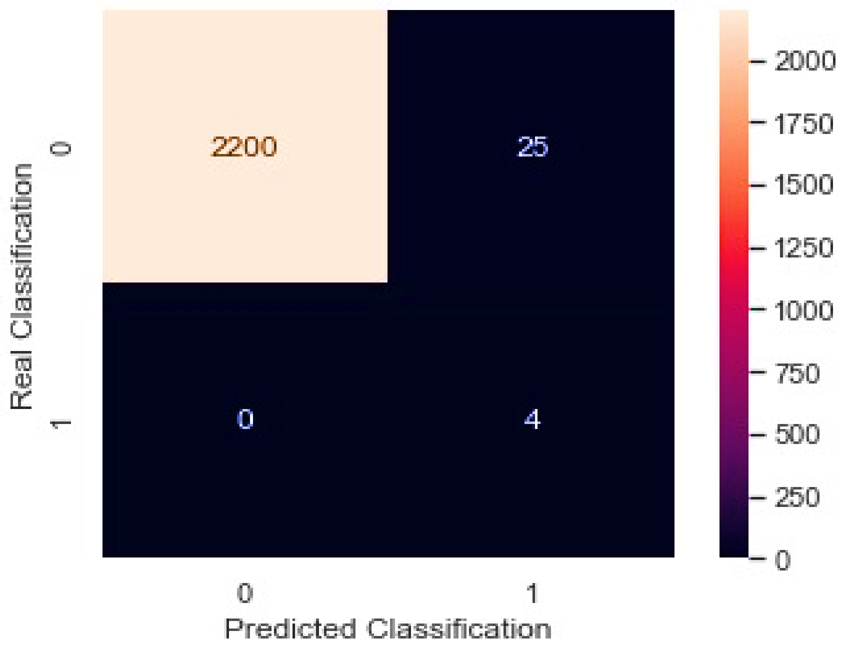


Figure 3.8 Confusion matrix visualization for LightGBM model

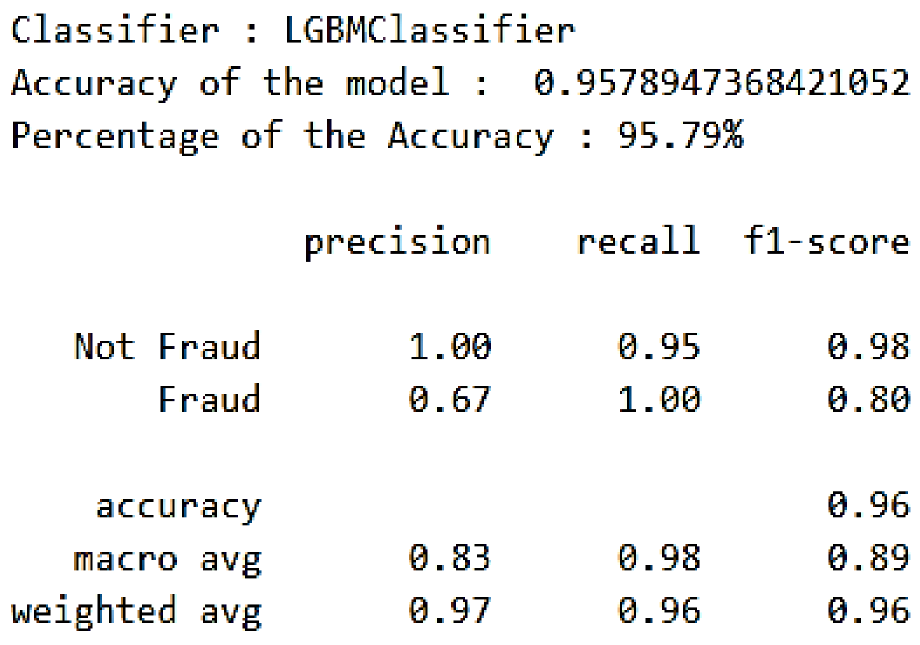
Conversely, it has misidentified fraudulent content as legitimate content 0 times and 25 times misidentified legitimate content as fraudulent. An important aspect of this could be discerned in Table 3.8

Table 3.3.1 - Classification Report of LightGBM classifier

Content Classification	F1 score	Recall	Precision	Support
Not Fraud	0.98	0.95	1.00	2200
Fraud	0.80	1.00	0.67	29

Above Fig 3.3.1 depicted the classification report of the optimum model which indicates the classifier's accuracy as 95.79% as above said.

Analyzed several scientific approaches and have proposed a novel solution as an intelligent and advanced web application that incorporates Machine Learning (ML) and Deep Learning (DL) technology, mainly aiming to surpass present-day resembling competitors and provide customers with the finest user experience when browsing online classifieds. As above said results have proven the best approach of detecting fraud content for car classifieds using machine learning appliances.



```
Classifier : LGBMClassifier
Accuracy of the model :  0.9578947368421052
Percentage of the Accuracy : 95.79%

              precision    recall  f1-score

   Not Fraud       1.00      0.95      0.98
       Fraud       0.67      1.00      0.80

    accuracy                           0.96
   macro avg       0.83      0.98      0.89
weighted avg       0.97      0.96      0.96
```

Figure 3.9 Classification report of Optimum LightGBM classifier

3.2 Research Findings

In this section, the above-mentioned testing's study findings are assessed given the discoveries of the existing literature, and this approach for amendment extension is offered and executed.

- The content detection researches revealed in the existing literature triggered the experimentation of several forms of classification tasks using different databases. literature study, it was conceivable to explore how detection performs with diverse machine learning algorithms such as Random Forest, Decision Tree, and Support Vector Machines in the incipient phases. The background of fraud detection investigations aided in producing the optimum solutions employing these algorithms, nevertheless the boosting algorithms such as LightGBM, XGBRF, and XGB for fraudulent content identification in classifieds advertisement matter have never been thoroughly tested.

- Nevertheless, As indicated in the preceding sections, the maximum efficiency was obtained using a classification matrix. Consequently, the LightGBM does have the highest precision, and the f1 score is one of the best metrics to measure efficiency, as per the research literature analysis. As a conclusion, we concentrated our efforts on achieving a high f1 score; as previously stated, the f1 score has a maximum value of 1 and a minimum value of 0. After conducting various iterations, it was determined that max depth 5 provides the greatest F1 score.

- Furthermore, it was discovered that employing a classified application's own ML-based process for the assessment of a fraud detection of specific advertisement content is being spammed was extremely beneficial for the application's internal functioning and the system developers' convenience.

- The literature review aided in the selection of stemming over lemmatization since lemmatization provides more precise conclusions. Snowball, as a result, is more voracious than the Porter stemmer. Many of the features incorporated into the Snowball stemmer were driven by difficulties encountered with the Porter stemmer.

3.3 Discussion

From an application developer's perspective, they need to implement strict program logic to identify and prevent fraudulent advertisements from being submitted either by accident or deliberately, using verification and validation techniques. If those ads were submitted, in due course, shoppers may view erroneous ads and lose confidence in using the application for their needs prompting the downgrade of the platform's reputation. Current applications do not prioritize the above, let alone monitor and prevent them from occurring.

Even more, many implemented classified advertising systems simply do not exhibit rich User Interfaces (UI s) for smooth functionality or promote quality user experience, in consonance with the latest trending scientific breakthroughs. Although since the execution of the industry research fundamental objectives and goals without losing time gratuitously, a publicly available dataset called 'Used cars dataset,' which had 441,396 second-hand car advertisement records extracted from Classified ads, must have been eventually received for the detecting fraud procedure such as in this investigation existing literature since the dataset was more than actually necessary. The dataset's intricacy was extensively identified during the data acquisition and exploratory analysis processes.

Nonetheless, this study used all records for model training and testing. Furthermore, most classified ads sites have not used complex machine learning algorithms (ML) in their applications production release. Due to the insufficient computational resource and since the data set being considered is comprises a high amount of memory storage data set future reduced to accomplish the task. As a result, this research developed the capability assumption, which was proven valid as a solution since that objective was eventually realized. After scrutinizing the prevailing issues, the authors have investigated. Another reason why this research is significant and novel when compared to comparable current research is that the majority of them have only assessed the algorithm detecting performances by examining them separately with the essential hyperparameters.

Additionally, boosting algorithm learning has never been used, particularly in automobile fraud classifieds ad content. Furthermore, For the client-server development, Angular-integrated services handle databases and API service requests from the front-end application, while the components manage the UI elements and their vivid, accurate responses even without repetitive page refreshing. It effectively functions as a back-end alternative, requiring no further programming. As the model backend implementation was properly constructed using flask microservice. Python was identified as the main technology for implementing the API, and the Flask microframework was used because of its appropriateness for creating APIs. Nevertheless, it was eventually discovered that Flask itself can not handle HTTPS request responses using CORS headers. As a consequence Flask plugin (Quart) was apprehended, as well as the API source code architecture was rebuilt.

In conclusion, the models in the API may collect car data from the UI advertising form that the consumer finishes out, prior to submitting and provide the response with the ultimate forecasted value as a warning or success to the client application for a presentation. Moreover, At the completion of all key component development, the model's deployment included APIs and all associated dependent files in a cloud environment. The droplets, obtained from the IaaS platform DigitalOcean, redundancy, scalability, and adaptability for the APIs and also the Tievs front-end applications. Existing research concentrating on fault context detection won't control or coordinate such plans of action. Not only should the engine establishing aspect be treated with the utmost respect, but then so does the deployment aspects if all of those components are to perform their primary functions without disruptions inaccessibility. Nevertheless, such models and APIs would indeed be ineffective, which is why this study intended to test that hypothesis and was successful.

3.4 Summary of Each Student's contribution

Table 3.2 IT18089400 Contribution Summary

Member	Component	Tasks
Ravihari J.M.S	Intelligent Classified Fraud Context Detection System	• Congregate the appropriate data set using external sources, or survey using craigslist region and an initial data records should be labeled as fraud and non-fraud inability. • Exploring and collection of the most convenient dataset that could be used for model training. • Preprocessing the dataset is required, as is exploratory data analysis, in order to successfully complete the data preparation process and feed the model. • Explore the most suitable machine learning algorithm considering the accurateness of the output speed or training time. • Comparative analysis using classification matrix and exposed the optimum model. • Hyperparameter tweaking for optimization for the revealed optimum model. • Fraud content detection Flask API implementation, Embedding the Mode importing the model pickles with the application without inconveniences. • Build client and server-side components and extract the data from the advertisement form, prior to posting. • Visualization on the front end with a warning message prior to posting, if the advertisement is consisting of inappropriate content.

4 CONCLUSION

Classified advertisement interactions have risen in popularity in comparison to its predecessor's publications, television, and broadcast communications networks. Along with the popularity of the classifieds sites as mentioned in the study question above, using verification and validation techniques to detect and prevent the publication of misleading advertising, whether inadvertently or intentionally, is a major undertaking. Fraudulent classifieds ads, websites masquerading as reputable sources of information, commodities, products, and services are proliferating, resulting in thousands of dollars in losses.

Due to the numerous adverse consequences of Online deception and fraudulent activities, multiple research and techniques have been established to detect counterfeit web classifieds applications; nevertheless, none of them have been able to provide adequate and appropriate responses to suppressing these fraudulent activities. From the perspective of an application developer, rigorous program logic must be implemented to detect and prevent fraudulent advertising from being published, either deliberately or inadvertently, employing acceptance testing approaches. If those advertisements were published, consumers may eventually see misleading ads and forfeit trust in utilizing the application for their requirements, causing the platform's reputation to degrade.

Current applications do not prioritize, much alone analyze and prevent the aforementioned events. This study offers a fraudulent content detection methodology on car classifieds prior to submitting the system based on content analysis of an advertisements' text descriptions, natural language processing, and supervised machine learning approaches. Publicly available Kaggle datasets were being used for the required ML model, while initially targeting the US market. The suggested approach is divided into four major phases: data acquisition, preprocessing, feature extraction, and classification, best model selection, hyperparameter tweaking, correct most accurate predictions about fraudulent activities and user interface, integration production deployment. To preserve scope, this study focused exclusively on used vehicle classifieds. Following EDA, the Data preparation approach was being used to minimize any disturbance from inputs that would otherwise impede the ability to differentiate between negative and positive classifiers. Considering Because of the processing power, the Stemmer method was used as a stemmed approach. Additionally, to operate, raw text was effectively transformed into several vectors, thus the BOW model was established to extract characteristics from the text. Following the deployment of the TF-IDF NLP vectorization approach, text input is assessed for relevance and serialized for further processing.

Succeeding competence of seven Machine learning algorithms after feature extraction every classifier was trained and tested in the above-mentioned ratio until the best model was achieved. The experimental observations revealed that such developed fraudulent advertisement contents detection model produced satisfactory accuracy and served the purpose of the investigation. Following the selection of LightGBM as the forecasting solution, depth-wise growth and proficiency to obtain random grid selection with the most relevant parameters were undertaken. The classification method was conveniently embedded in the REST API to assess ad content as authentic or not, and Tievs automatically displayed near advertising submission when necessary.

ML model and front-end app API development, programming languages, and frameworks such as Python, Angular were utilized, along with supporting Angular and python libraries. More convenient development IDEs Ides were utilized as technical tools and test REST APIs and resuscitate APIs in the event of a hosting error, whereas PostgreSQL has been used as a database server. Perhaps adjusting the supervised learning with a large fraudulent content dataset, to boost recall rate precisely while decreasing accuracy loss, might be accomplished in the foreseeable future works using deep learning architecture.

Also, because the identification process was exceedingly imbalanced, there have been fewer fraud/spam cases in the populace if we have randomly chosen a subset of instances for evaluation. In an attempt to overcome the issue, want to use the active learning approach in future experiments.

5 REFERENCE

[1] H. Tran, T. Hornbeck, V. Ha-Thuc, J. Cremer, and P. Srinivasan, “Spam detection in online classified advertisements,” *ACM Int. Conf. Proceeding Ser.*, vol. 11, Apr. 2011, doi: 10.1145/1964114.1964122.

[2] M. Maktabdar Oghaz, A. Zainal, M. Maarof, and M. Kassim, *Content based Fraudulent Website Detection Using Supervised Machine Learning Techniques*. 2017.

[3] Z. Gyongyi and H. Garcia-Molina, “Web Spam Taxonomy,” presented at the First International Workshop on Adversarial Information Retrieval on the Web (AIRWeb 2005), Chiba, Japan, Apr. 2005, Accessed: Feb. 26, 2021. [Online]. Available: http://ilpubs.stanford.edu:8090/771/.

[4] A. McCormick and W. Eberle, “Discovering Fraud in Online Classified Ads,” p. 6.

[5] “Agile Methodology: What is Agile Software Development Model?” https://www.guru99.com/agile-scrum-extreme-testing.html (accessed Feb. 26, 2021).

[6] “Six Steps to Master Machine Learning with Data Preparation,” *KDnuggets*. https://www.kdnuggets.com/six-steps-to-master-machine-learning-with-data-preparation.html/ (accessed Feb. 26, 2021).

[7] A. Ntoulas, M. Najork, M. Manasse, and D. Fetterly. “Detecting spam web pages through content analysis. In WWW '06: Proceedings of the 15th international conference on World Wide Web, pages 83{92, New York, NY, USA, 2006. ACM.

[8] Ntoulas, A., Hall., B., Najork, M., Manasse, M. and Fetterly, D.: Detecting Spam Web Pages through Content Analysis. In Proceedings of *15th Int. Conf. World Wide Web*, pp. 83–92 (2006).

[9] Shen, G., Gao, B. Liu, T. Y., Feng, G., Song, S. and Li, H.: Detecting link spam using temporal information. In Proceedings IEEE Int. Conf. Data Mining, ICDM, no. 49, pp. 1049–1053 (2006).

[10] Becchetti, L., Donato, D., Baeza-yates, R. and Leonardi, S.: Link Analysis for Web Spam Detection. ACM Transactions on the Web.vol. 2 no. 1, pp. 1-42 (2007).

[11] Drost, I. and Scheffer, T.: Thwarting the nigritude ultramarine: Learning to identify link spam. Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 3720 LNAI, pp. 96–107 (2005).

[12] Abbasi, A.: Detecting Fake Medical Web Sites Using Recursive Trust Labeling. In Proceedings 14th Int. Worldw. Web Conf., vol. 30, no. 4, pp. 1-22 (2012).

[13] Nguyen, H.T. and Smeulders, A. "Active Learning Using Pre-clustering", Proceedings of the 21st International Conference on Machine Learning, 2004, pp. 623-630,http://www.aicml.cs.ualberta.ca/_banff04/icml/pages/papers/94.pdf.

[14] Kaufman, L. and Rousseeuw, P.J., "Partitioning Around Medoids", Finding Groups in Data, Wiley-Interscience, 2005, pp. 68-125.

[15] Erande, Ameya S., and Alok K. Verma. "Measuring agility of organizations-a comprehensive agility measurement tool (CAMT)." International Journal of Applied Management and Technology 6.3 (2008).

[16] ADITI. (2005). “Agile Methodology based Services,” [Online] Available at: http://www.aditicorp.com/services/agile-methodology-based-services/ [Accessed Feb. 22, 2021].

[17] B. Settles, “Active Learning Literature Survey,” p. 47.

[18] A. McCormick and W. Eberle, “Discovering Fraud in Online Classified Ads,” p. 6.

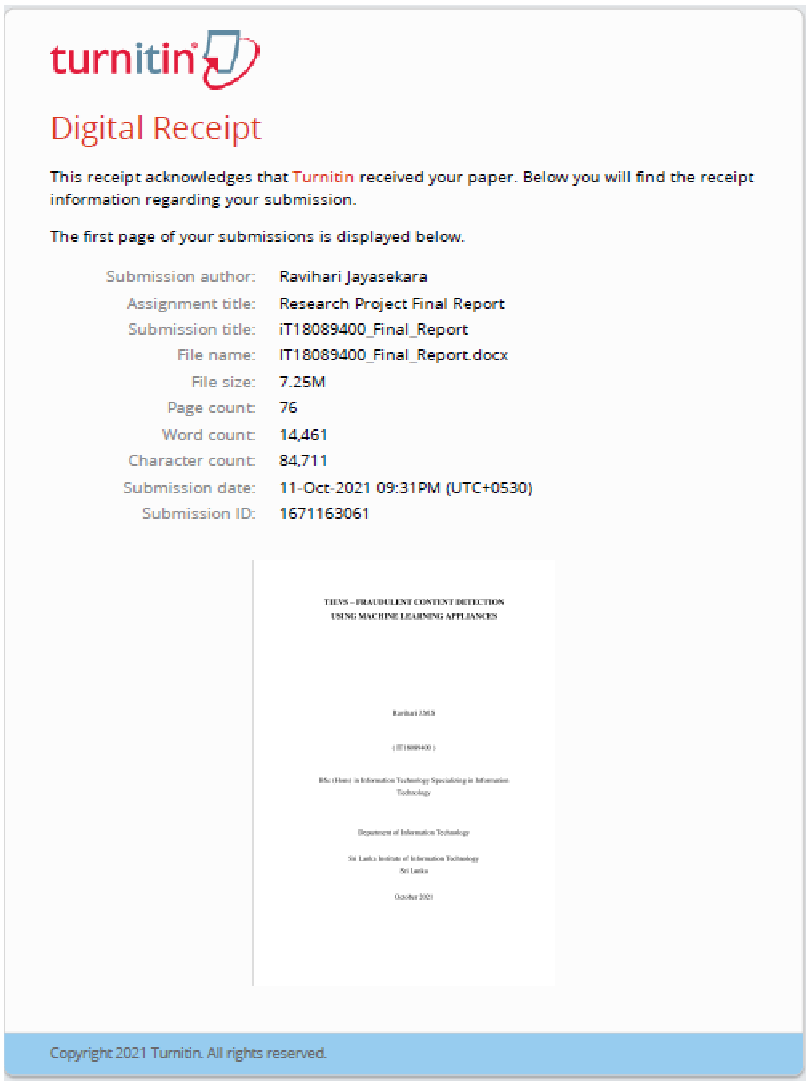
[19] V. Garg and S. Nilizadeh, “Craigslist Scams and Community Composition: Investigating Online Fraud Victimization,” in *2013 IEEE Security and Privacy Workshops*, San Francisco, CA, USA, May 2013, pp. 123–126. doi: 10.1109/SPW.2013.21.

[20] D. Fetterly, M. Manasse, and M. Najork, “Spam, damn spam, and statistics: using statistical analysis to locate spam web pages,” in Proceedings of the 7th International Workshop on the Web and Databases colocated with ACM SIGMOD/PODS 2004 – WebDB 04, Paris, France, 2004, p. 1. doi: 10.1145/1017074.1017077.

[21] H. Tran, T. Hornbeck, V. Ha-Thuc, J. Cremer, and P. Srinivasan, “Spam detection in online classified advertisements,” ACM Int. Conf. Proceeding Ser., vol. 11, Apr. 2011, doi: 10.1145/1964114.1964122.

6 APPENDIX

Appendix A: Turnitin Plagiarism Report Receipt



turnitin

Digital Receipt

This receipt acknowledges that Turnitin received your paper. Below you will find the receipt information regarding your submission.

The first page of your submissions is displayed below.

Submission author:	Ravihari Jayasekara
Assignment title:	Research Project Final Report
Submission title:	iT18089400_Final_Report
File name:	IT18089400_Final_Report.docx
File size:	7.25M
Page count:	76
Word count:	14,461
Character count:	84,711
Submission date:	11-Oct-2021 09:31PM (UTC+0530)
Submission ID:	1671163061

TIEVS – FRAUDULENT CONTENT DETECTION USING MACHINE LEARNING APPLIANCES

Ravihari J.M.S

(IT18089400)

BSc (Hons) in Information Technology Specializing in Information Technology

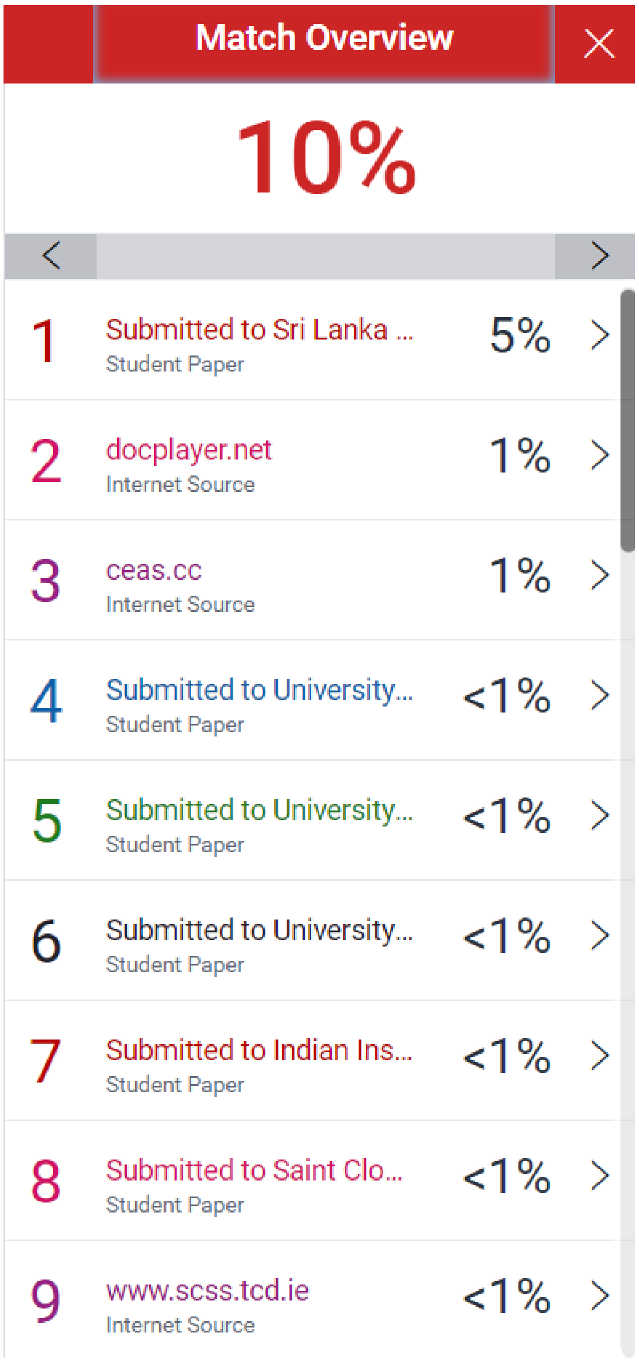
Department of Information Technology

Sri Lanka Institute of Information Technology
Sri Lanka

October 2021

Copyright 2021 Turnitin. All rights reserved.

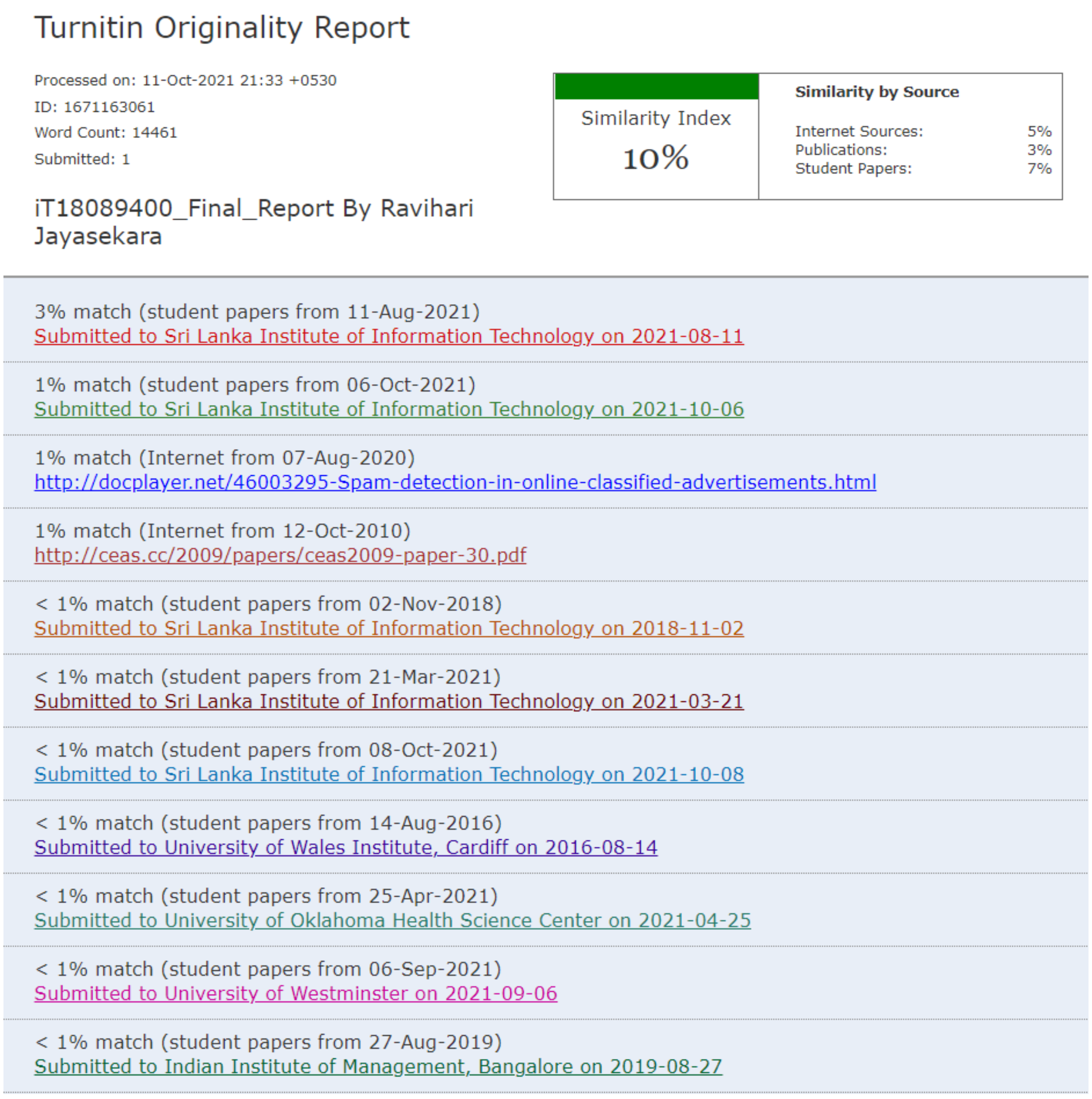
Appendix B: Turnitin Plagiarism Report



Match Overview

10%

1	Submitted to Sri Lanka ...	Student Paper	5%
2	docplayer.net	Internet Source	1%
3	ceas.cc	Internet Source	1%
4	Submitted to University...	Student Paper	<1%
5	Submitted to University...	Student Paper	<1%
6	Submitted to University...	Student Paper	<1%
7	Submitted to Indian Ins...	Student Paper	<1%
8	Submitted to Saint Clo...	Student Paper	<1%
9	www.scss.tcd.ie	Internet Source	<1%



Turnitin Originality Report

Processed on: 11-Oct-2021 21:33 +0530
ID: 1671163061
Word Count: 14461
Submitted: 1

iT18089400_Final_Report By Ravihari Jayasekara

Similarity Index	Similarity by Source	
10%	Internet Sources:	5%
	Publications:	3%
	Student Papers:	7%

3% match (student papers from 11-Aug-2021)
Submitted to Sri Lanka Institute of Information Technology on 2021-08-11

1% match (student papers from 06-Oct-2021)
Submitted to Sri Lanka Institute of Information Technology on 2021-10-06

1% match (Internet from 07-Aug-2020)
http://docplayer.net/46003295-Spam-detection-in-online-classified-advertisements.html

1% match (Internet from 12-Oct-2010)
http://ceas.cc/2009/papers/ceas2009-paper-30.pdf

< 1% match (student papers from 02-Nov-2018)
Submitted to Sri Lanka Institute of Information Technology on 2018-11-02

< 1% match (student papers from 21-Mar-2021)
Submitted to Sri Lanka Institute of Information Technology on 2021-03-21

< 1% match (student papers from 08-Oct-2021)
Submitted to Sri Lanka Institute of Information Technology on 2021-10-08

< 1% match (student papers from 14-Aug-2016)
Submitted to University of Wales Institute, Cardiff on 2016-08-14

< 1% match (student papers from 25-Apr-2021)
Submitted to University of Oklahoma Health Science Center on 2021-04-25

< 1% match (student papers from 06-Sep-2021)
Submitted to University of Westminster on 2021-09-06

< 1% match (student papers from 27-Aug-2019)
Submitted to Indian Institute of Management, Bangalore on 2019-08-27

< 1% match (student papers from 09-May-2020)
Submitted to Saint Cloud State University on 2020-05-09

< 1% match (Internet from 10-Mar-2020)
https://www.scss.tcd.ie/publications/theses/diss/2019/TCD-SCSS-DISSERTATION-2019-001.pdf

< 1% match (publications)
Hossein Fereidooni, Mauro Conti, Danfeng Yao, Alessandro Sperduti. "ANASTASIA: ANdroid mAlware detection using STatic analySIs of Applications", 2016 8th IFIP International Conference on New Technologies, Mobility and Security (NTMS), 2016

< 1% match ()
Hirei, Hassan Mohamed. "Investigating and Validating Scam Triggers: A Case Study of a Craigslist Website", theRepository at St. Cloud State, 2020

< 1% match (Internet from 19-Apr-2021)
https://tutorsonspot.com/questions/questions-on-logistic-regression-using-r-owsm/

< 1% match (Internet from 09-Jun-2020)
https://www.coursehero.com/file/57327732/IT16000940-2pdf/

< 1% match (student papers from 19-Sep-2016)
Submitted to University of Southampton on 2016-09-19

< 1% match (Internet from 26-Jul-2021)
https://acp.copernicus.org/articles/20/11065/2020/

< 1% match (Internet from 24-Mar-2016)
http://www.ccsenet.org/journal/index.php/ijbm/article/download/52016/31078

< 1% match (student papers from 12-Mar-2019)
Submitted to University of Stellenbosch, South Africa on 2019-03-12

< 1% match ()
Han, Li. "Mechanical behaviour of self-piercing riveted aluminium joints", University of Hertfordshire, 2003

< 1% match (Internet from 30-Jan-2014)
http://www.andrew.cmu.edu/user/jgernand/dissertation/Gernand_Dissertation_Pocket.pdf

< 1% match (student papers from 23-Aug-2021)
Submitted to University College London on 2021-08-23

< 1% match (Internet from 06-May-2016)
http://airccj.org/CSCP/vol5/csit54403.pdf

< 1% match (Internet from 29-Aug-2017)
http://www.dl.kuis.kyoto-u.ac.jp/webquality2011/p35-tran.pdf

< 1% match (Internet from 23-Jul-2018)
https://repository.tudelft.nl/islandora/object/uuid:efe4605c-a677-4756-8040-44957a602bab/datastream/OBJ/download

< 1% match (Internet from 11-Sep-2003)
http://202.145.96.30/download/splus-arrayanalyzer/splus-aa-whitepaper.pdf

< 1% match (publications)
Hamad Alsaleh, Lina Zhou. "A Heuristic Method for Identifying Scam Ads on Craigslist", 2018 European Intelligence and Security Informatics Conference (EISIC), 2018

< 1% match (Internet from 15-Jul-2021)
https://calhoun.nps.edu/bitstream/handle/10945/55598/17Jun_Funk_Daniel.pdf

< 1% match (Internet from 30-Sep-2020)
http://theiet.lk/wp-content/uploads/2017/10/Proceedings-Final-23RD.pdf

< 1% match (publications)
"Proceedings of the Future Technologies Conference (FTC) 2020, Volume 1", Springer Science and Business Media LLC, 2021

< 1% match (student papers from 01-Sep-2020)
Submitted to Higher Education Commission Pakistan on 2020-09-01

< 1% match (publications)
Sheng, Zhang, Chen Hailong, Jiang Chuan, and Zhang Shaojun. "An adaptive time window method for human activity recognition", 2015 IEEE 28th Canadian Conference on Electrical and Computer Engineering (CCECE), 2015.

< 1% match (student papers from 22-May-2021)
Submitted to University of Bedfordshire on 2021-05-22

< 1% match (Internet from 05-Mar-2013)
http://aran.library.nuigalway.ie/xmlui/bitstream/handle/10379/2674/paper.pdf?sequence=1

< 1% match (Internet from 06-Aug-2020)
https://pubmed.ncbi.nlm.nih.gov/30257926/

< 1% match (publications)
Hung Tran, Thomas Hornbeck, Viet Ha-Thuc, James Cremer, Padmini Srinivasan. "Spam detection in online classified advertisements", Proceedings of the 2011 Joint WICOW/AIRWeb Workshop on Web Quality - WebQuality '11, 2011