King Hussein School of Computing Sciences

Computer Science Department

13477 Software Engineering

Instructor

Dr. Ammar Odeh

Heba Najdawi	Rania Al-Ali	Rakan Armoush	Sara Selwadi
20190959	20190388	20190009	20190975

The Github repository (including the notebook and the dataset)

https://github.com/rakanarmoush1/SE-Project

Table of Contents

1. Introduction

Phishing is a word for identity thieves' persuading attempts of obtaining personal information from a pond of unsuspecting Internet users. Data breaches now cost companies $4.24 million per incident on average [1]. Phishers try to obtain personal information and financial accounts' details such as usernames and passwords by using fake websites and phishing software. This research looks into different strategies for detecting phishing websites using machine learning techniques to analyse various features of benign and phishing URLs (Uniform Resource Locator). Machine learning is an important branch of artificial intelligence that is defined as a machine's ability to imitate intelligent human behaviour. So instead of having someone manually determine whether a specific website is a phishing website, which is time consuming and impractical, machine learning will help us achieve this faster and more accurately. We look at how host features, lexical features, and page importance properties are utilised to detect phishing websites. In order to gain greater understanding of the URLs' structures that endorse phishing, we explore multiple different algorithms for the evaluation of features. An algorithm is a procedure or a sequence of rules to follow to solve a problem or output a result of the desired type. The fine-tuned parameters aid in the selection of the most appropriate machine learning algorithms for distinguishing between phishing and benign sites.

2. System description

The system's main function is detecting any possible phishing attack through URLs. The proposed system uses Catboost as its ensemble algorithm to solve this supervised machine learning problem. supervised machine learning refers to the using labeled datasets in order to train algorithms that will accurately classify previously unseen data into two discrete values; as detecting phishing attacks is a binary classification problem. After feeding our input data into the model, the weights will be altered and tuned until the model is suitably fitted, which takes place in the cross validation phase. The dataset used to train these models consists of over 500,000 records (URLs) [2]. The system will train the models using 80% of the dataset, then test the accuracy of the model on the other 20%.

3. System purpose

Phishing is a technique that uses social engineering to steal sensitive information from users. It is a major threat to all internet users and most of the time, it works by tricking them into visiting a website that's full of malicious content and most of these phishing sites look identical to real websites, therefore machine learning models and statistical methods have been widely used in various applications to handle phishing attacks; by either preventing the user from entering the website or notifying the user if they have visited a malicious website.

The proposed system aims to detect phishing attacks using machine learning models, statistical methods, and feature engineering by extracting useful features from URLs (i.e. whether a URL is an IP address or not). An optimal solution that we aim to use in our research to detect phishing involves having the lowest computational complexity and the highest possible f1-score which is the core metric of evaluating our machine learning model.

4. Problem statement

Phishing attacks are breaches that can allow attackers to steal people's private and sensitive information by the use of suspicious URLs, the attacker can lure the users into giving information such as usernames, passwords, credit card numbers, etc. and can sometimes lead to money or identity theft. The following figure demonstrates a typical phishing scenario

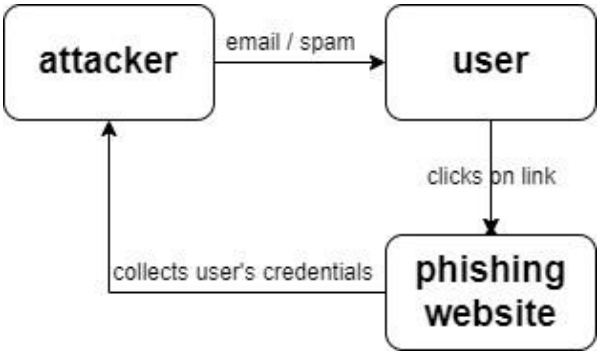

Figure 1: phishing attack scenario

APWG reported that the number of detected attacks per month was between 68,000 and 94,000 with 24.9% of all phishing attacks being against financial organizations and payment providers [3].

In order to decrease those numbers more efficient detection systems are needed, and this paper proposes an ensemble machine learning approach that can be utilized for more accurate attacks' predictions.

5. System context view

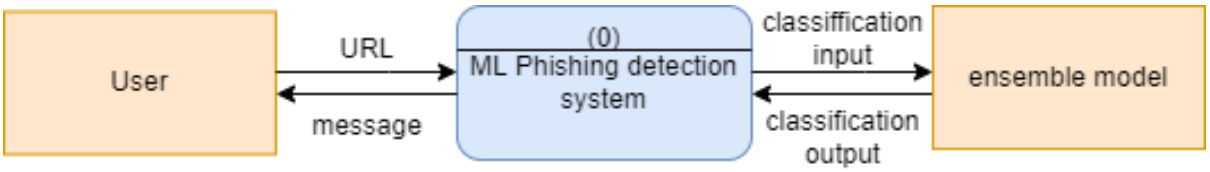

Figure 2: the system context view

6. Literature review

This part of the paper looks at other researcher's approaches at using ensemble techniques to detect phishing in order to compare results and evaluate our approach.

Ubing et al. used a dataset about phishing websites, that had 5126 data points and 30 features [4], after applying feature selection algorithms, several machine learning models were implemented individually (averaged perceptron, Bayes point machine, boosted DT, Decision jungle, Deep SV, logistic regression, NN, and SVM) as well as an ensemble learning model that used these multiple models together to predict the final outcome. The boosted decision tree achieved the best accuracy of 97.3% but the proposed ensemble model achieved the best recall score.

another ensemble learning technique is stacking, which combines the predictions of various machine learning models into one final model, this technique was tested on phishing classification by Vinitha et al. in their research [5], the model proposed a stacking classifier of XGBoost, Random Forest, and Naïve Bayes, which was trained on a two-class dataset of the size 2905. The proposed stacking algorithm achieved an accuracy of 85.6%.

Abdulhamit et al. also used an ensemble learning approach in their research [6], the data they used in their work was a public phishing websites dataset provided by the UCI repository of machine learning which included 2456 instances, the classification was implemented using multiple models individually, combining these models with MultiBoosting, and with Adaboost. The best accuracy was achieved by Adaboost with SVM which was 97.61% and an f-measure of 0.976
The approach of using an extreme gradient boosting(XGBoost) classifier with random forest and K-NN as the base classifiers was proposed by Jiaqi Gu and Hui Xu [7], fine hyper-parameter tuning was done on the model. The dataset used in this research was balanced and consisted of 11054 data points with each data point being classified as one of two classes (Phishing or Non-phishing), their proposed model outranked the K-NN model that was tested in the paper with an accuracy of 96.4% and an f-measure of 0.96, but the random forest model performed better than the proposed model with an accuracy 96.7% and an f-measure of 0.968

Alsariera et al. proposed the use of multiple classifiers and tree-based algorithms [8] (NBTree and BFTree) with bagging, Adaboost, and MABoost ensemble techniques, and then a 10-fold cross validation was used to resample and evaluate the ensemble models. The models were trained on three different phishing datasets (A:11055 variaables, B:10000 variables, c:1353 variables) with data sets A and C showing slight imbalance and dataset B being balanced. The best accuracies achieved on the three datasets (A, B, C) were 97.09% by the BoostedBFTree model, 98.38% by the BoostedNBTree model, and 90.61% by BaggedNBTree respectively. In general bagging, boosting, and multiBoosting recorded better results than the other individual classifiers.

Al-Sarem et al. [9] also used three medium sized website phishing datasets (1000-31000 variables in each dataset) with datasets 1 and 3 presenting some imbalance and dataset 2 being balanced, the paper proposed the use of genetic algorithm-based ensemble classifiers which were XGBoost, Random Forests, Bagging, Adaboost, LightGBM, and GradientBoost. Random Forests reported the best results in the first and the third dataset, with an accuracy of 97.02% and 97.15%, respectively,

and f-measures of 0.9749 and 0.9590 respectively. LightGBM reported an accuracy of 98.65% in the second dataset and an f-measure of 0.9865, which was the best, it also achieved the best precision, recall.

Basit et al. used a balanced website phishing dataset with 30 features and 11055 records that was provided by the UCI machine learning repository in their research [10]. The proposed classification method was to use the random forest classifier and combining it with three different classifiers (ANN, C4.5, and K-NN) using ensemble majority voting, the ensemble algorithm of K-NN combined with RFC attained the accuracy of 97.3% and the f-measure of 0.976, which was the best amongst all other models.

7. Challenges

- Due to most URLs being benign, the challenge of dealing with imbalanced data surfaces. Imbalanced data is commonly used to evaluate the distribution of observations in a given set of data. For instance, one class has a high number of observations while the other one has a low number. Hence depending on accuracy as the only evaluation metric will decrease the reliability of the model, so by resorting to other evaluation metrics, such as precision and recall, the results will be more reliable.
- One of the biggest challenges machine learning faces is the lack of data. Detecting phishing URLs using machine learning also faces this challenge, having enough data for the machine learning model to learn from is essential for finding hidden patterns that would be harder to find otherwise.
- Overfitting the training data is also a challenge when training a machine learning model. This is due to the low number of records used for training, and hyper-parameter tuning.

8. Projection

The table below shows the tasks of phase 1.

Table 1: phase 1 tasks

Activity	Predecessors	Duration (days)
Finding the idea and splitting tasks	-	1
Introduction	-	2
System design	Introduction	2
System purpose	Introduction	2
Challenges	-	1
Past work (literature review)	-	2
Gantt chart	-	2

Network diagram	Gantt chart	2
Context view	System design, system purpose	2

Gantt chart

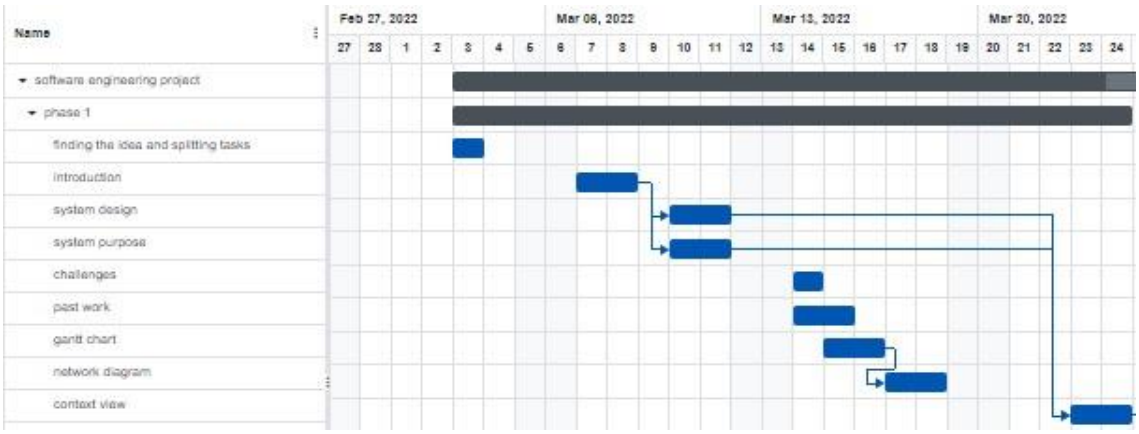

Figure 3: phase 1 Gantt chart

Network diagram

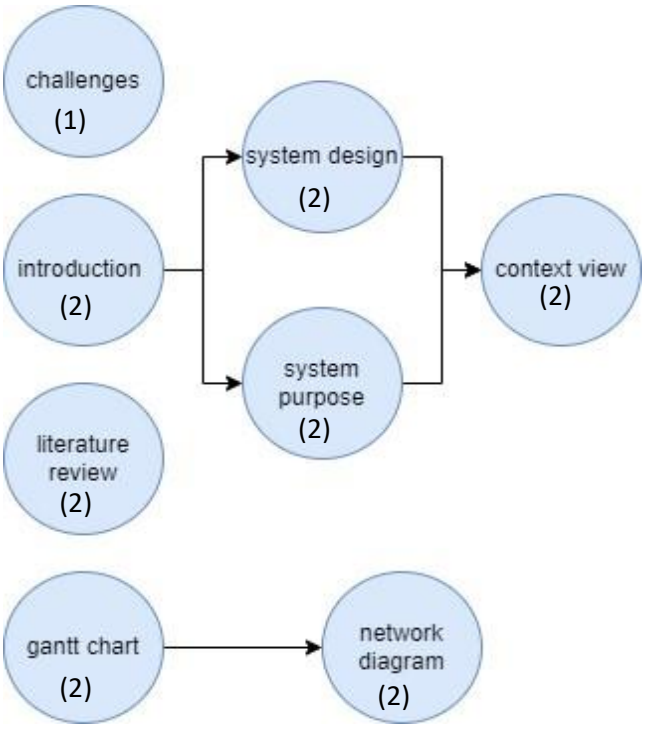

Figure 4: network diagram for phase 1

9. Functional requirements of the system

Table 2: Functional Requirements for phase 2

Requirement ID	Requirement Name	Requirement description
FR1	URL Entry	The system must accept a URL as input from user
FR2	URL Classification	The system must be able to classify whether a URL is benign or not by passing it to the ensemble model

10. Non-Functional properties of the system

Table 3: Non-Functional Requirements for phase 2

Requirement ID	Requirement Name	Requirement description
NFR1	Accuracy & Performance	Measuring the performance of the ensemble model by finding the confusion matrix of the test data and computing measures such as Fmeasure.
NFR2	Hyperparameter Tuning	Tuning the hyperparameters of each model to increase accuracy and performance by using techniques such as random search and grid search
NFR3	Valid URLS	The system must be able to determine whether a URL is valid through previously defined rules.
NFR4	Availability	The system must be available anytime.
NFR5	Usability	The user interface of a system should be easy to use.

11. Class diagram

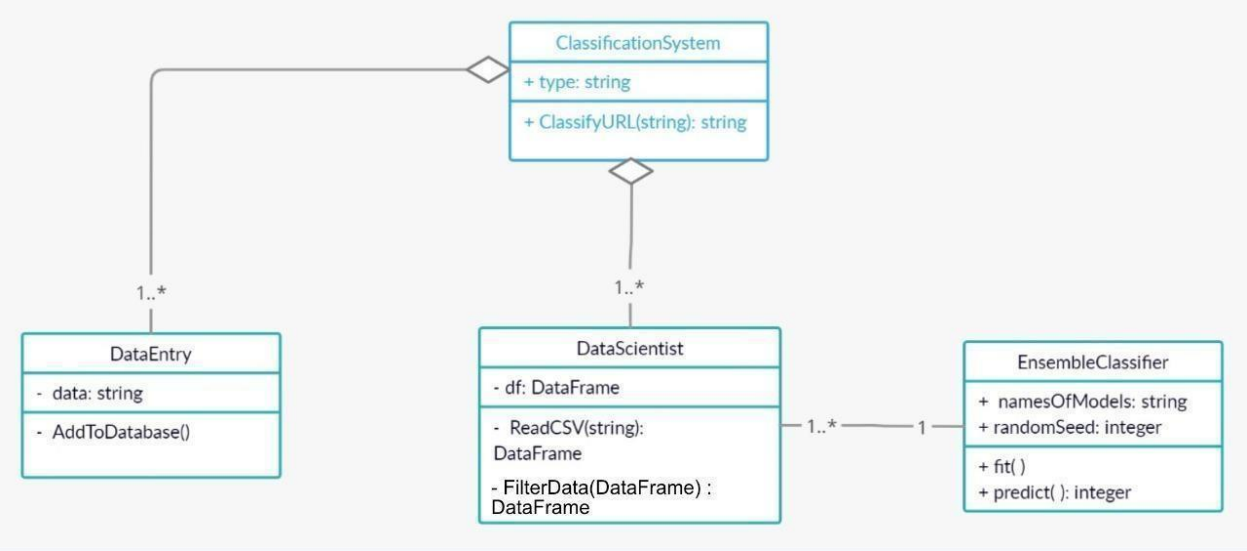

Figure 5: Class Diagram for phase 2

Shown above is the class diagram for how the system acts when it is given the dataset to train the classification system, first the data scientist reads the URL dataset, and passes it to the ensemble classifier, then the classifier fits the dataset and returns a model ready to classify any given URL with the predict function that returns 1 or 0 (1 for phishing 0 for benign).

12. Use case diagrams

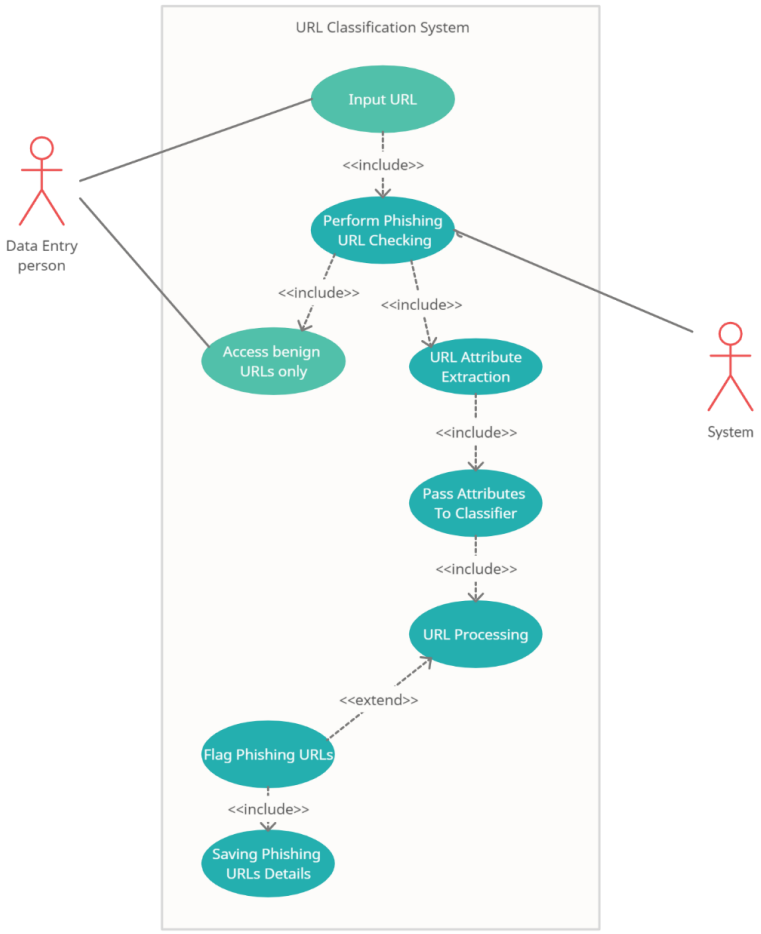

Figure 6: Use Case Diagram 1

- Use Case: Input URL
- Actors: Data Entry Person
- Purpose: To allow the user to enter the URL that they wish to classify
- Overview (Success scenario):
  Having the URL successfully entered into the classification system
- Type: Primary
- Cross References: FR1
- Typical course of actions: The Data Entry Person and the System. The Data Entry Person inputs the URL into the URL classification system and the system begins to check the URL. The system extracts the URL's attributes and passes it to the classifier to process it. If the URL is benign, the system gives the data entry person access to it. If the URL is a phishing URL, it could be flagged as a phishing URL and have its details saved.

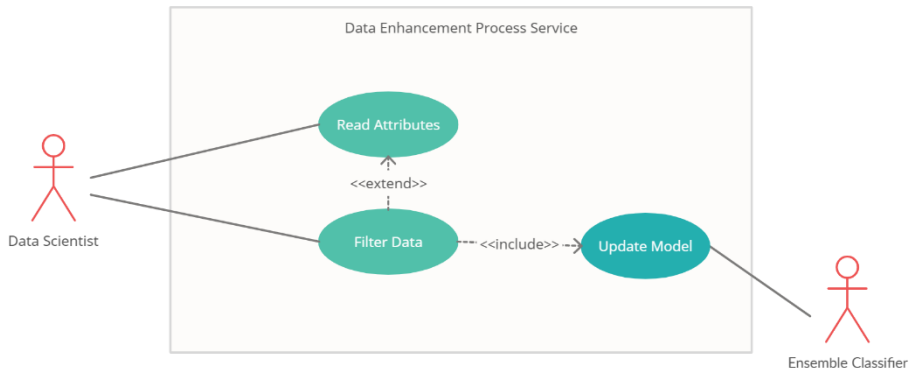

Figure 7: Use Case Diagram 2

- Use Case: Read Attributes
- Actors: Data Scientist
- Purpose: Extracting useful features from URL
- Overview (Success scenario): Extracting features that increase model accuracy and performance
- Type: primary
- Cross References: FR1, NFR1, NFR3
- Typical course of actions: This how the ensemble classifier can update the model using the help of a data scientist. When the data scientist reads the attributes, they can filter data in order to help the ensemble classifier update the model.

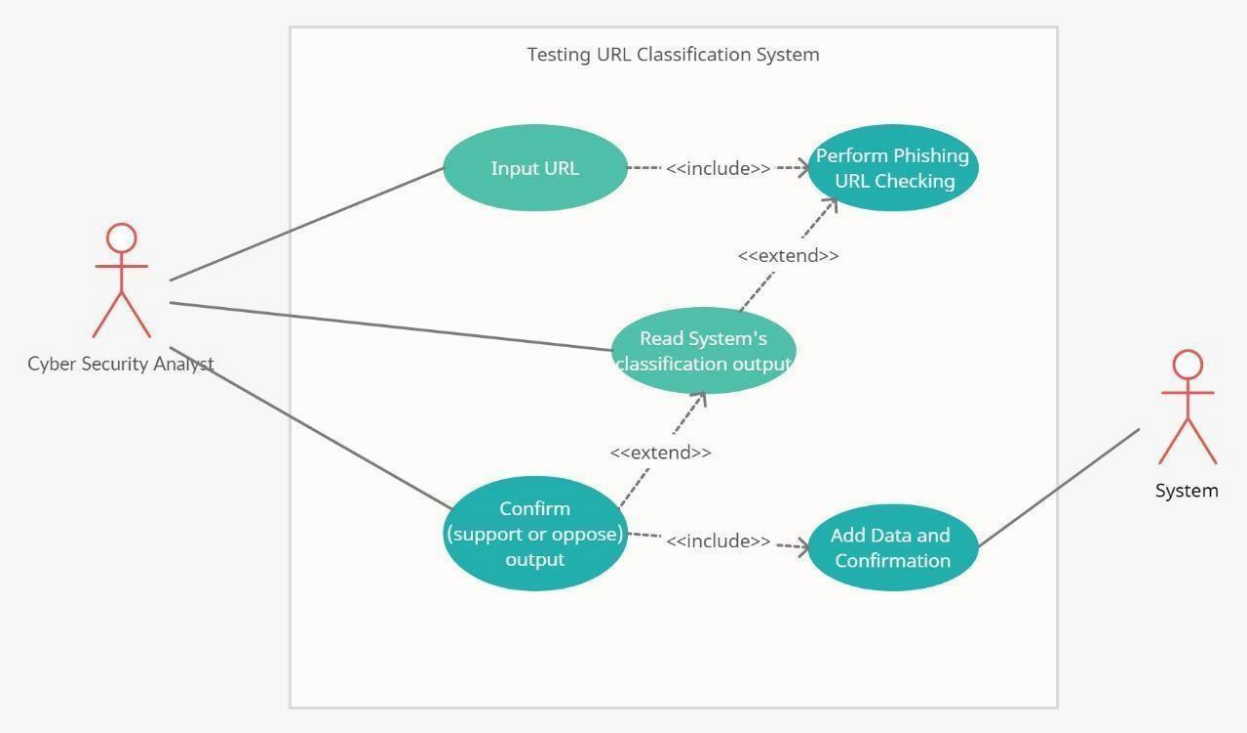

Figure 8: use case diagram 3

- Use Case*:* Confirm (support or oppose) output
- Actors: Cyber Security Analyst
- Purpose: to double check that the classifier's output is correct based on the knowledge of the Cyber Security Analyst
- Overview (Success scenario): Having the Cyber Security Analyst confirm or oppose the output of the model in a correct way
- Type: Primary
- Cross References: FR2, NFR1
- Typical course of actions: During the process of testing the system, a cyber-security analyst or professional inputs a URL the way a data entry person would. However, when the system checks the given URL and classifies it, the cyber security analyst is capable of determining whether the classification given by the system is correct or not.

13. Activity diagrams

Shown below is the activity diagram for how the system acts when the user receives a URL.

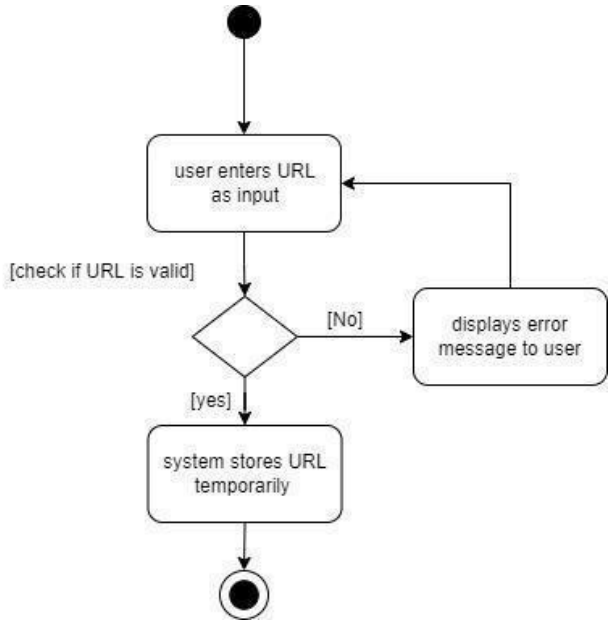

Figure 9: activity diagram 1

The user will enter a URL as an input to the system, the URL would be stored temporarily in the system's inputs list until it's classified as either a phishing or benign.

The figure below is the activity diagram for the main detection system

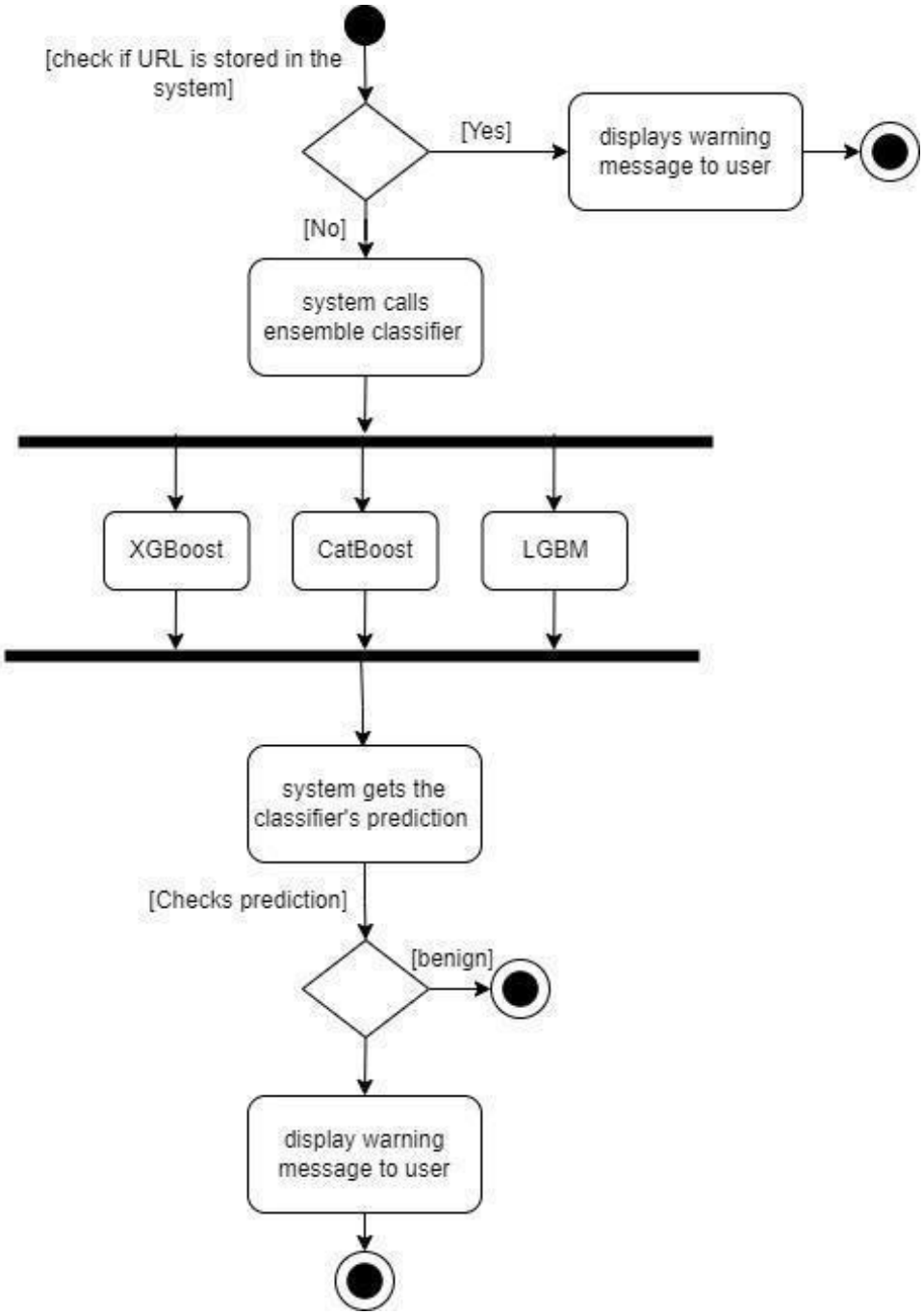

Figure 10: activity diagram 2

The system will check if the URL is stored in the system's phishing websites' list, if so the user will be notified with a warning message to prevent them from visiting the website.

If the URL was not seen before by the system, the ensemble classifier will be called (using the best models after testing the system), and the link will be classified as either phishing or benign. If the website is classified as a phishing website, the user will be notified with a warning message.

Below is the activity diagram for how the system acts after classifying a website.

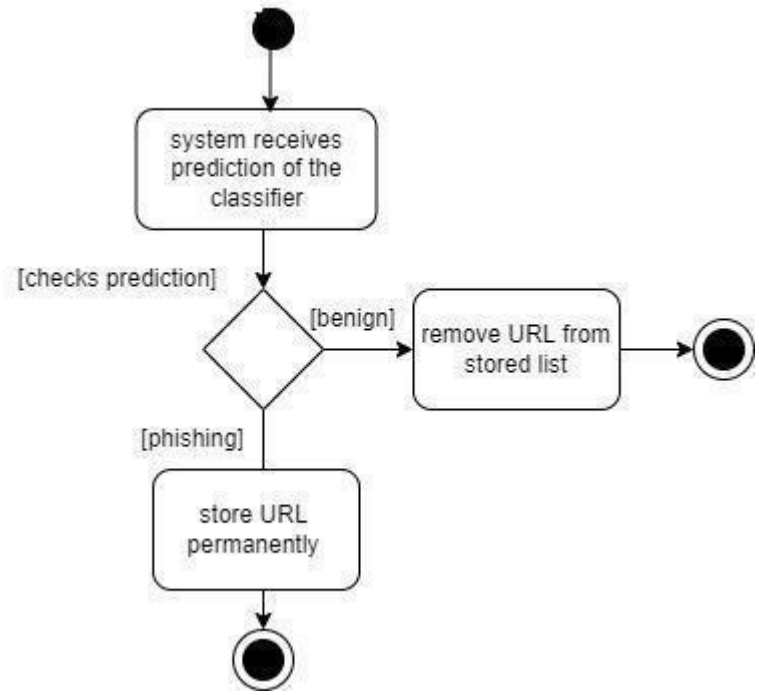

Figure 11: activity diagram 3

After calling the classifier and labelling the URL, the URL will be removed from the temporary inputs' list.

If the URL is classified as a phishing website, it will be moved to the permanent list of phishing links so that if the user enters the same URL again, they will be warned right away without calling the classifier.

If the website is classified as a benign website, it will not be stored in the system.

14. Sequence diagrams

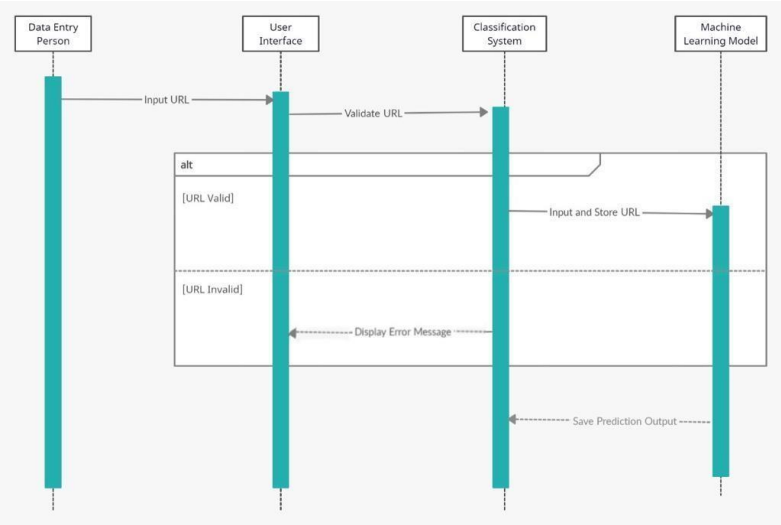

Figure 12: sequence diagram 1

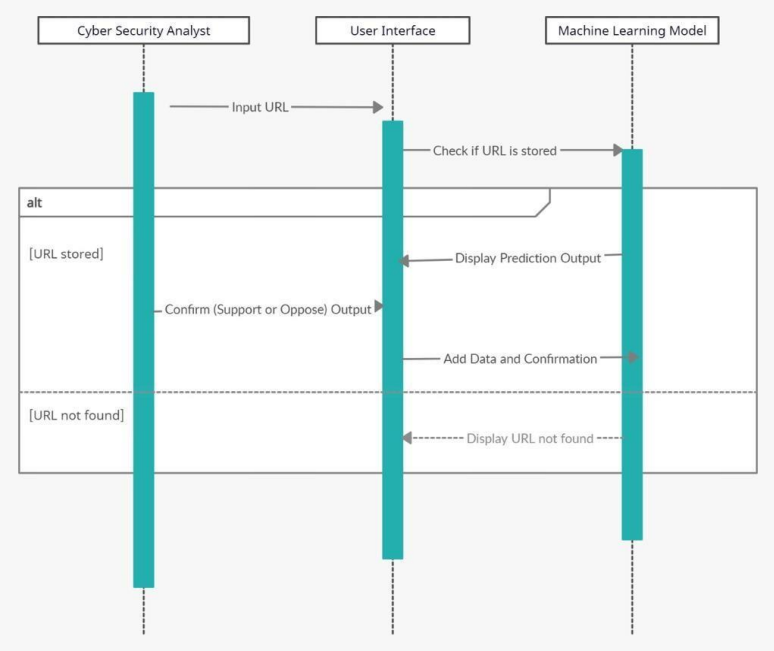

Figure 13: sequence diagram 2

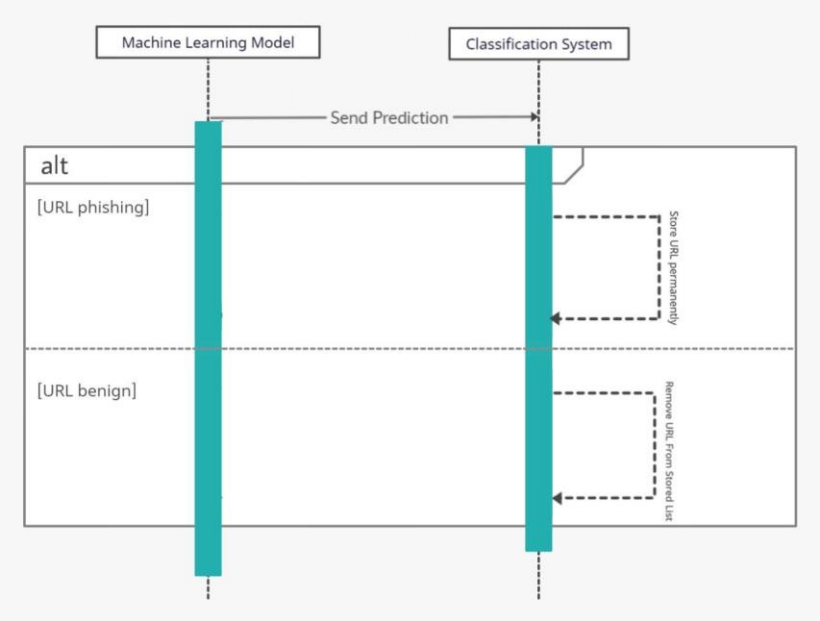

Figure 14: sequence diagram 3

15. Proposal Algorithm

The purpose of this paper is developing an ensemble machine learning model that can classify URLs into phishing or benign (safe) URLs. The proposed system is supposed to take any URL as an input and accurately predict the class. In this research we aim to find the effectiveness of using three different gradient boosting algorithms which are CatBoost, XGBoost and LightGBM.

Our system architecture consisted of three main stages: 1-data preparation; data collection, data cleaning, and feature engineering. 2-learning process; training our proposed ensemble models on the training subset of the data, parameter tuning, and validation. 3-evaluation; using our evaluation metrics.

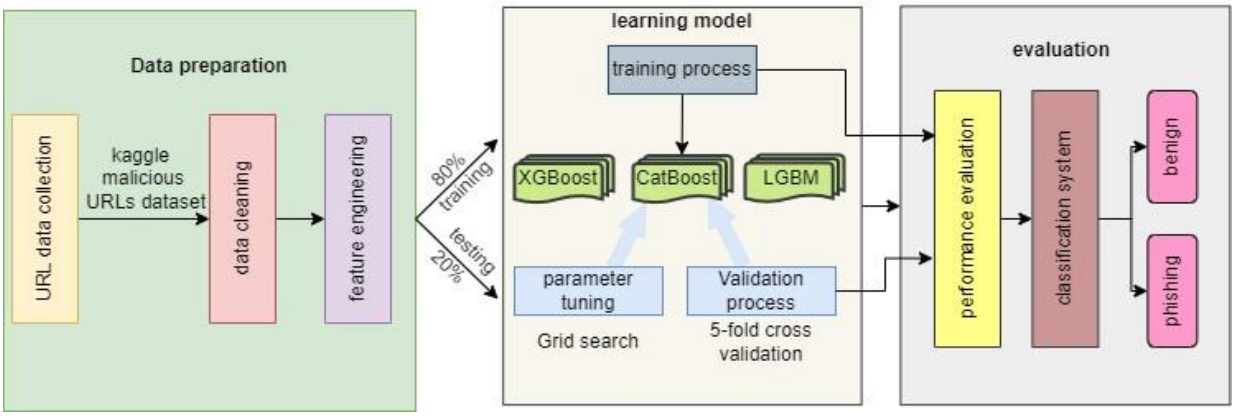

Figure 15: the architecture of the proposed framework

The dataset used in this research was the Kaggle malicious URLs dataset, which included 522214 URLs, the counts of the URLs belonging to each class is shown in that table below.

Table 4: counts of each class in the dataset

	BENIGN	PHISHING
TRAIN	342488	75283
TEST	85615	18828
TOTAL	428103	94111

The dataset included no features. In order to make it easier for the model to detect patterns more efficiently we performed feature engineering. The final dataset consisted of 18 engineered features.

The following table shows the engineered features with their description

Table 5: features' description

Feature	Description
is_ip	Indicates whether a URL is an IP address
contains_shortener	Indicates whether a URL used a shortening service
url_len	Length of URL
subdomain_len	Length of subdomain in the URL
tld_len	Length of top-level domain in the URL
fld_len	Length of free level domain in the URL
url_path_len	Length of the URL's path
url_alphas	Number of letters in the URL
url_digits	Number of numbers in the URL
url_puncs	Number of punctuation marks in the URL
count.	Count of . (dots) in the URL
count@	Count of @ in the URL
count-	Count of - in the URL
count%	Count of % in the URL
count?	Count of ? in the URL

count=	Count of = in the URL
count_dirs	Count of the directories in the URL
first_dir_len	Length of the first directory in the URL

Figure 16 shows the correlation heatmap between each pair of the engineered features which uses the Pearson correlation coefficient formula shown below.

$$\frac{\sum(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum(x_i - \overline{x})^2 \sum(y^i - \overline{y})^2}}$$

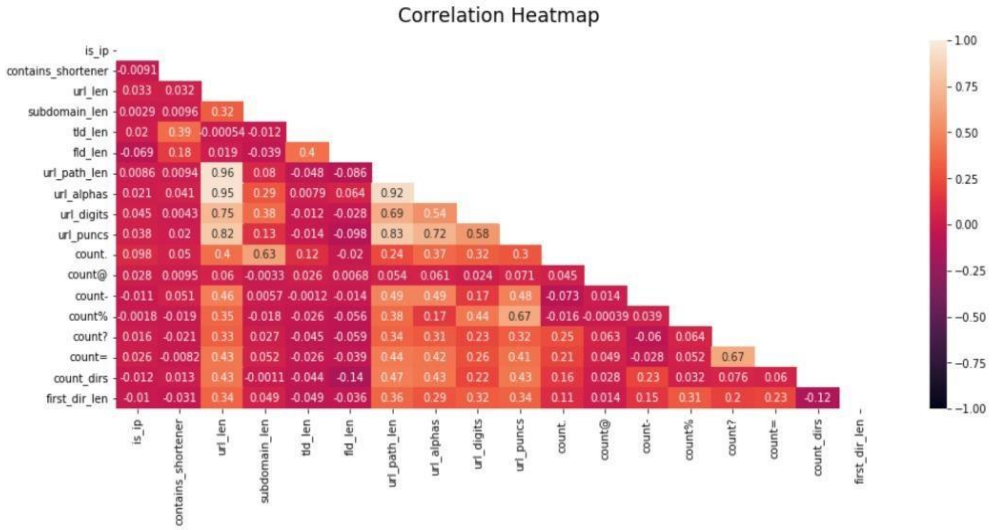

Figure 16: Pearson correlation heatmap

After preparing the data, the dataset was split into two subsets according to the Pareto 80/20 principle; training (80%) and testing (20%). The three proposed models were all gradient boosting algorithms.

A single model of decision tree was implemented as a form of unit testing in order to compare the ensemble model with it and determine if our proposed model was successful and performed better than the single model.

Boosting is an ensemble machine learning method that uses decision trees individually and combined sequentially in order for the model to produce better predictions.

We used boosting algorithms because of their ability to learn from each previous model's mistakes by giving weights to each previous prediction. Furthermore, boosting solves classification problems in an accurate and a fast way. the three used algorithms were:

a) Catboost

Catboost is a machine learning algorithm that uses gradient boosting on decision trees [11]. First, we used the default hyperparameters of the model and then after using grid search and cross validation to find the best hyperparameters for the problem, we set the learning rate to 0.1, max depth to 10 which means that each decision tree will have a maximum depth of 10.

b) XGBoost (Extreme gradient boosting)

XGBoost is defined as an optimized distributed gradient boosting algorithm [12]. It carries out machine learning algorithms under the Gradient Boosting framework. XGBoost delivers a parallel tree boosting technique that is supposed to be extremely flexible, portable, fast and efficient. As for the hyperparameters we set the learning rate to 0.1, and the max depth to 3.

c) LightGBM (Light gradient boosting machine)

LightGBM is a gradient boosting framework that applies tree based learning algorithms [13]. It speeds up the training process, provides higher efficiency, and is able of handling large scale datasets. As for the hyperparameters we set the learning rate to 0.1, and the max depth to -1 which means that there will be no limit on the depth of each tree.

A 5-fold cross validation is used to reduce bias and make sure that the model performs well with any subset of testing data.

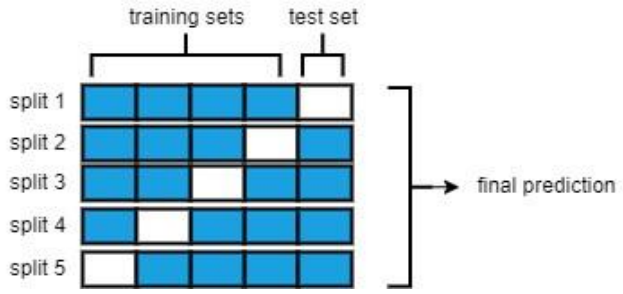

Figure 17: visual representation of a 5-fold cross validation process

After training the model, binary classification will be carried out on the testing data, and the performance of the model will be evaluated using the following evaluation metrics: accuracy, f-measure, precision, and recall.

The measures we used are calculated using the following equations.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F - Measure = 2\,\frac{Precision \times Recall}{Precision + Recall}$$

In which TP stands for true positive, FP stands for false positive, TN stands for true negative, and FN stands for true negative.

16. Simulation Result

Our research revealed that the ensemble model performed better than the traditional techniques. The single model used was a decision tree model and it achieved an accuracy of 95%, an f-measure of 0.97, recall of 0.97, and a precision of 0.97. below is the confusion matrix of the decision tree model.

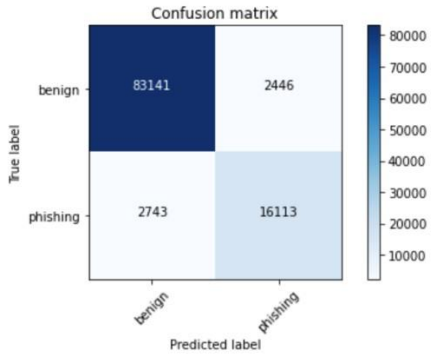

Figure 18: confusion matrix of decision tree model

In this work, we developed an ensemble machine learning model for detecting whether a URL is benign or phishing. To measure the effectiveness of the proposed approach, we have evaluated the system performance in terms of the evaluation metrics mentioned above, and we report our empirical results in this section.

Table 6: performance summary of proposed models

	CatBoost	**XGBoost**	**Light GBM**
Precision	0.98	0.93	0.96
Recall	0.98	0.97	0.98
Accuracy	96.9%	92.1%	95.2%
F-measure	0.98	0.95	0.97

The table above shows that the CatBoost model was the best proposed model with an accuracy of 96.9% and an F-measure of 0.98, so this model can be the one selected from this research to predict detect phishing websites.

Figure 19 below shows the feature importance in the CatBoost model.

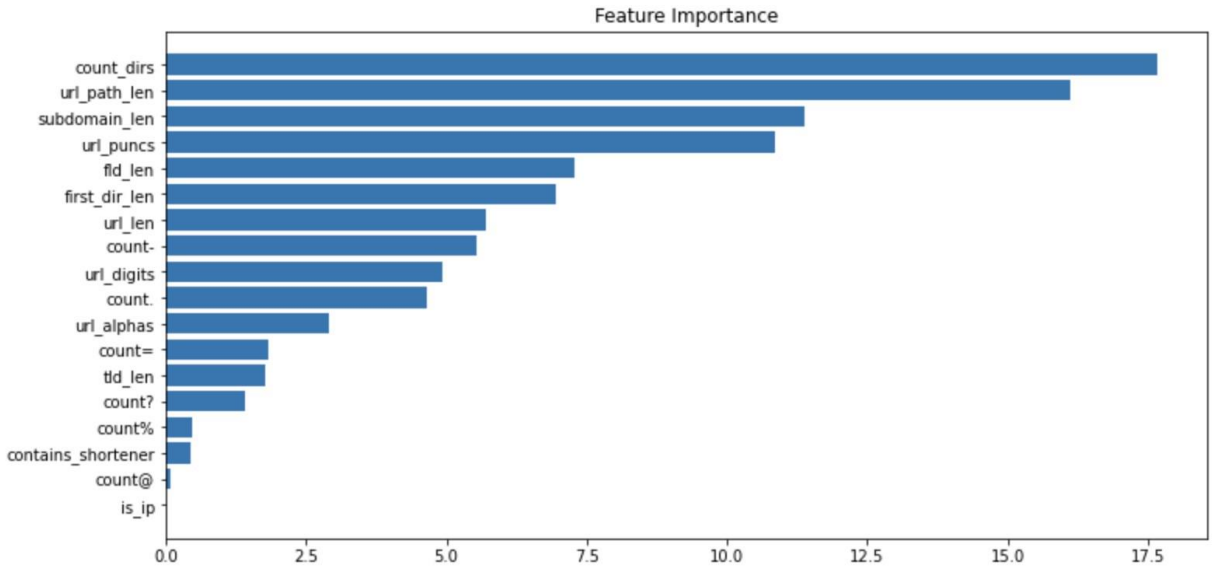

Figure 19: feature importance of features in the CatBoost model

Initially, figures 20 through 22 show the confusion matrices for the developed ensemble models when implemented on the testing data. The figures demonstrate the high robustness of the models since the number of correctly classified URLs is much higher than the number of falsely classified ones. For example, the total of correctly classified URLs (true positives and true negatives) in the CatBoost model was 101237 and the total of incorrectly classified URLs (false positives and false negatives) was 3206 out of 104443. The same pattern is shown in the confusion matrices of the other developed models.

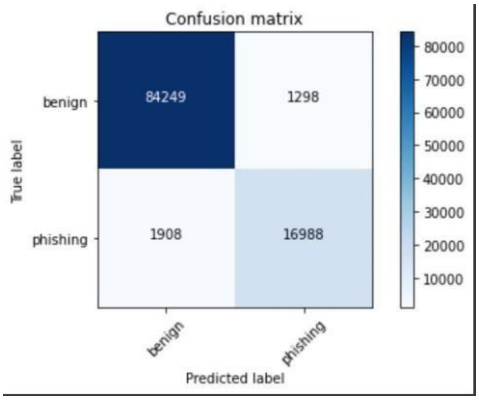

Figure 20: confusion matrix for CatBoost model

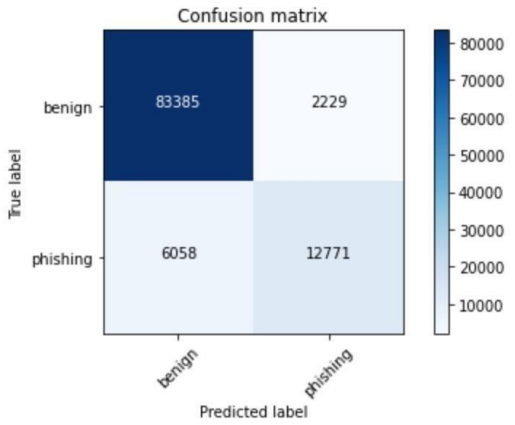

Figure 21: confusion matrix for CatBoost model

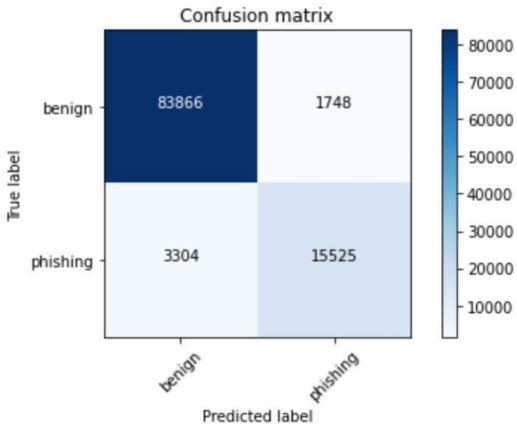

Figure 22: confusion matrix for CatBoost model

17. Comparison

Table 7: comparison of related works and proposed algorithm

Reference number	Model Year	Classification Model	F-measure	Accuracy	Dataset size
[4]	2019	Boosted decision tree	0.97	97.3%	5126
[5]	2021	Stacking Model: (XGB, RF, Naïve Bayes)	-	85.6%	2905
[9]	2021	LightGBM	0.98	98.7%	10,000

[6]	2020	Adaboost with SVM	0.97	97.6%	2456
[10]	2020	Ensemble algorithm(KNN combined with RFC)	0.97	97.3%	11,055
[8]	2022	Boosted-NBTree	0.98	98.4%	10,000
[7]	2022	Random Forest	0.96	96.4%	11,054
Proposed	2022	CatBoost	0.98	96.9%	522,214

By using the F-measure score as our main measure to evaluate the performance of a model, we assess the model based on the true positives and negatives hence solving the class imbalance problem; For example, when the class ratio is 99:1 and the model keeps on predicting the first class it will get 99% accuracy without paying attention to the second class.

18. Conclusion and future work

In summary, this paper proposed the use of gradient boosting learning in the field of detecting phishing websites; three models were developed and tested. The proposed models (CatBoost, XGBoost, LGBM) were tested on the Kaggle malicious URLs dataset after performing feature engineering. The job of the models was to provide a binary classification (either phishing or benign) for the given URLs. We investigated the use of these models in many different ways in order to achieve the best results, and we tried to increase the efficiency of the models by the use of grid search for parameter tuning and cross validation to reduce the bias of the models.

The CatBoost, XGBoost, LGBM models resulted in the accuracies of 96.9%, 92.1% and 95.2% respectively and the f-measures of 0.98, 0.95, 0.97 respectively. Overall the CatBoost model achieved the best results amongst the other proposed models in this paper.

Our proposed model did good compared to other researches who used boosting, considering that the dataset used in our research was much bigger than any of the referenced related works.

Our main goal in the future is to develop a complete system with greater accuracy and low bias. We intend to test our best model on different datasets in order to further evaluate it and to improve its overall performance.

In addition, the system can be improved and developed to also classify some URLs as suspicious websites, in order to warn the users and decrease the false negative rates.

References

1. (2021). IBM Report: Cost of a Data Breach Hits Record High During Pandemic.
2. Siddharth, M. Malicious URLs dataset, Version 1. Retrieved March 03, 2022 from https://www.kaggle.com/datasets/sid321axn/malicious-urls-dataset.
3. (2021). APWG: Phishing Activity Trends Report Q3 2021. *Computer Fraud & Security*.
4. Ubing, A.A., Jasmi, S.K., Abdullah, A.B., Zaman, N., & Supramaniam, M. (2019). Phishing Website Detection: An Improved Accuracy through Feature Selection and Ensemble Learning. *International Journal of Advanced Computer Science and Applications*.
5. V. K. Nadar, B. Patel, V. Devmane and U. Bhave, "Detection of Phishing Websites Using Machine Learning Approach," *2021 2nd Global Conference for Advancement in Technology (GCAT)*, 2021, pp. 1-8
6. Subasi, A., & Kremic, E. (2020). Comparison of Adaboost with MultiBoosting for Phishing Website Detection. *Procedia Computer Science, 168*, 272-278.
7. J. Gu and H. Xu, "An Ensemble Method for Phishing Websites Detection Based on XGBoost," *2022 14th International Conference on Computer Research and Development (ICCRD)*, 2022, pp. 214-219,
8. Alsariera, Yazan & Balogun, Abdullateef & Adeyemo, Victor & Tarawneh, Omar & Mojeed, Hammed. (2022). INTELLIGENT TREE-BASED ENSEMBLE APPROACHES FOR PHISHING WEBSITE DETECTION. Journal of Engineering Science and Technology. 17. 563-0582.
9. Al-Sarem, M., Saeed, F., Al-Mekhlafi, Z.G., Mohammed, B.A., Al-Hadhrami, T., Alshammari, M.T., Alreshidi, A., & Alshammari, T.S. (2021). An Optimised Stacking Ensemble Model for Phishing Websites Detection. *Electronics, 10*, 1285.
10. Basit, A.W., Zafar, M., Javed, A.R., & Jalil, Z. (2020). A Novel Ensemble Machine Learning Method to Detect Phishing Attack. *2020 IEEE 23rd International Multitopic Conference (INMIC)*, 1-5.
11. https://catboost.ai/
12. https://xgboost.readthedocs.io/en/stable/
13. https://lightgbm.readthedocs.io/en/latest/