CS-C3240: Machine Learning

Project report: Phishing Websites Detection

2024-10-06

Contents

1 Introduction

Phishing, a pervasive cybercrime, has significantly escalated its attacks on the digital landscape.As an increasing number of users rely on online platforms for banking, government services, and other critical activities, their sensitive information, including identification numbers, usernames, passwords, and banking credentials, is at risk of exposure to malicious actors. The ease with which counterfeit websites can be created, mirroring the appearance and content of legitimate ones, makes users particularly susceptible to inadvertently entering their personal data into fraudulent domains. To mitigate this threat and safeguard internet users from phishing attacks, a robust phishing detector or classifier is essential. Leveraging the power of machine learning algorithms, the phishing detector employed in this project effectively categorizes websites as either *Phishing* or *Legitimate*.

The report is structured as the following: Section 2 formulates the problem by introducting the data set, specifically labels and features. Section 3 explains feature selection and introduces chosen machine learning models. Section 4 shows the final results of proposed models and compares the performance between different approaches. Section 5 presents conclusions based on the results shown in Section 4.

2 Problem Formulation

2.1 Dataset

The Phishing Dataset contains 5000 instances of phishing webpages and 5000 of legitimate webpages which were collected from January to May 2015 and from May 2017 to June 2017 [7]. By leveraging the browser automation framework, Selenium WebDriver, extracted features are more precise and robust compared to the regular-expression-based approaches [1]. All webpages in the dataset originates from the following sources:

- Phishing webpage: PhishTank, OpenPhish
- Legitimate webpage: Alexa, Common Crawl

This dataset is WEKA-ready, which means that it is in a format compatible with the WEKA[1] machine learning workbench.

2.2 Label

In this project, I will develop a machine learning (ML) model that could detect whether a given website is a phishing or not. Each instance in the dataset is labeled as a phishing or a legitimate webpage by a value of 1 or 0 respectively at the *label* column. In other words, the aim of this project is to develop a supervised machine learning model to classify a given website into two classes, *Phishing* and *Legitimate*, based on selected features obtained from the dataset.

2.3 Feature

There are 48 features in total, which most of them are the number of occurences of suspicious HTML elements or characters in URL or in web context, which are discrete values. Besides, there are continuous features which are the percentage of external hyperlinks, resources, redirects in the HTML webpage source code. Moreover, there are binary features indicating True/False boolean value as 1/0. For example:

- *DoubleSlashInPath*: Check if "//" exist in path of URL.
- *NoHttps*: Check if HTTPS exist in URL.

Additionally, there are categorical features which are encoded into three integer values: -1,0,1.

A full list of features could be investigated at [1][2].

[1] https://waikato.github.io/weka-wiki/
[2] See a summary of features at Table 5

3 Methods

3.1 Feature Engineering

As there are in total 48 features corresponding to only 10000 data poinst, it has pitfalls if the ML model is trained onto all features which number of them show weak correlationship to the labels. Additionally, with the fixed number of training samples, the predictive power of a ML model starts deteriorating when the number of dimensions, or features, exceeds a certain dimensionality [8]. Hence, keeping the number of features growing does mean increasing the accuracy of predictions.

To decide which subset of features I should rely on, the baseline features were referenced, showed in [1]. That baseline feature set were extracted by a feature selection framework introduced in [1]. Along with those 10 baseline features, the correlation coefficients between all features and label are analyzed by Mutual Information (MI). MI was chosen since it measures the general dependence between variables, both linear and non-linear relationship, which is unlike other common approaches such as Spearman or Pearson which only measure the linear dependence [4].Figure 1 shows MI scores of all 48 features sorted in the ascending order. Top 15 features which have the highest MI score were selected then were union-ed to the baseline feature subset to have the final feature sets for training. As a result, a set of 16 features was introduced as shown in Table 6. The number of features, or dimensions, is sufficient for the dataset of 10,000 entries and not too large leading to the curse of dimensionality.

3.2 Model

3.2.1 Logistic Regression

In this supervised classification task, Logistic Regression (LR) was proposed as a machine learning method since it is a binary classification and easily interpretable. LR categorizes data points by feature vectors $\mathbf{x} \in \mathbb{R}^n$ and binary lables y, $y \in \{0, 1\}$ (encoded to {Legitimate, Phishing}) in this case. The model uses a linear hypothesis $h(\mathbf{x}) = w^T\mathbf{x}$, with some parameter vector $w \in \mathbb{R}^n$, to predict a binary label y [3].

The binary logistic loss function was considered

$$L((\mathbf{x}, y), h) = log(1 + exp(-yh(\mathbf{x}))) \tag{1}$$

where x is feature vectors, y is the true binary label (0 or 1), and h is the linear hypothesis [3]. The reason behind this selection is that it is intuitive and already integrated into scikit-learn. Moreover, it penalizes incorrect predictions more heavily than correct predictions, which is suitable for the aim of the task of detecting phishing websites.

3.2.2 Random Forest

As shown in [1, 5], Random Forest (RF) outperforms other classifier in phishing detection. Additionally, unlike LR, RF does not rely on the linear relationship so that it is able to handle non-linear, more complex, relationships between features. Those were two main logically reasons leaded me to choose RF as a second machine learning model for this phishing dectection task. The loss function for RF in classification is the impurity of individual decision region for labels [3].

Regarding the splitting criteria, loosely understood as a loss function for decision trees, Gini impurity was selected due to its fast computation and good performance on balanced data set. Gini impurity measures the heterogeneity of a node, quantifying how often a randomly chosen element from the set would be incorrectly classified if it was labeled according to the distribution of labels in that node. Mathematically, Gini impurity is expressed:

$$I_G(p) = 1 - \sum_{i=1}^{K} p_i^2 \tag{2}$$

where K is the set of classes, p_i is the probability of choosing an item with label i [2]. The goal during training is to minimize the Gini impurity of the resulting child nodes, creating more homogeneous groups.

3.2.3 Model Evaluation

To evaluate the models, *Precision* score was measured as the main benchmark during hyperparameter-tuning. Additionally, in a phishing dectection context, it is also more important to dectect all phishing pages, even if some legitimate pages are misclassified, so the *Recall* metric was also considered during hyperparameter-tuning. Hence, *F1-Score* was also used to find balance between *Precision* and *Recall* metrics.

3.3 Training and Validation Set

The dataset itself is balanced between two classes, 5000 data poitns for each class, so class balancing step is no needed. Following the "thumbs-up" rule, the data set was divided into three sets, following the ratio 70/15/15, into three subsets: training, validation, and testing. As a result, the training, validation, and testing subset has 7000, 1500, and 1500 data points respectively. This ratio of splitting is reasonable as it

is widely used and accepted in the machine learning community. Moreover, this "hold-out" technique generates a sufficiently large size of training and validation data. Importantly, the data set was shuffled to avoid biased sampling. The training set was used to train the learning model, then the validation set was used for fine-tuning. If the current parameter and configuration of the hypothesis model has a good performance, final testing and benchmark are operated on the testing set.

4 Results

4.1 Logistic Regression

By running GridSearchCV with the grid parameters (regularization strength, norm of the penalty, and optimization algorithm), each with 5-fold cross-validation, I obtained the most optimal model with the following parameters: inverse of regularization strength 10, L1 penalty, and *liblinear* solver.

The optimal LR model generated the classification results shown in Table 1:

	Precision	Recall	F1-Score
Legitimate (0)	0.93	0.92	0.92
Phishing (1)	0.92	0.93	0.93
Accuracy			0.92

Table 1: Classification Report of the LR model on the validation set.

Generally, the LR had quite good performance on the validation set: all evaluation metrics were greater than 0.9. Moreover, *Recall* of the *phishing* class was greater than its *Precision*, which satisfied the importance of misclassified phishing websites.

4.2 Random Forest

GridSearchCV was employed to find an optimal RF model equipped with the set of hyperparameters, including the number of trees, and the maximum depth of the tree. The optimal RF model, along with 300 trees and no upper bound of depth[3], generated the classification results shown in Table 2.

	Precision	Recall	F1-Score
Legitimate (0)	0.99	0.98	0.98
Phishing (1)	0.98	0.99	0.98
Accuracy			0.98

Table 2: Classification Report of the RF model on the validation set.

All evaluation metrics produced by the optimal RF model were nearly perfect. Especially, the model did excellently on labeling all phishing websites since the *Recall* metric was 0.99, which means only 1% of phishing pages in the validation set bypassed the RF classifier.

[3]Then nodes are expanded until all leaves are pure or until leaves contain less the minimum number of samples at a leaf node [6].

4.3 Method Comparision

As depicted in Tables Table 1 and Table 2, the RF model exhibited substantially superior evaluation metrics (*Accuracy*, *Precision*, *Precision*, and *F1-Score*) compared to the LR model. This enhancement can be attributed to the RF model's capacity to capture intricate, nonlinear feature relationships that the linear LR model is unable to discern. Furthermore, the RF model demonstrated consistent performance on both the validation and test sets, as illustrated in Table 3. The negligible discrepancy between the metric scores on these two sets suggests that the RF model has effectively generalized to unseen data, indicating robust generalization capabilities.

As a result, the RF model is the final classifier for phishing detection.

	Precision	Recall	F1-Score
Legitimate (0)	0.98	0.98	0.98
Phishing (1)	0.98	0.98	0.98
Accuracy			0.98

Table 3: Classification Report of the RF model on the test set.

4.4 Feature Set Comparision

While maintaining the same parameter configuration, the model was trained with the baseline 10-feature set introduced in [1]. The evaluation results, presented in Table 4, demonstrate that the model achieved commendable performance even with the reduced feature set. Notably, the F1-Score and Accuracy both reached 0.97, only marginally lower than the scores obtained with the full custom feature set (F1-Score and Accuracy of 0.98).

	Precision	Recall	F1-Score
Legitimate (0)	0.97	0.96	0.97
Phishing (1)	0.96	0.97	0.97
Accuracy			0.97

Table 4: Classification Report of the RF model (baseline feature) on the test set.

5 Conclusion

In conclusion, both proposed models demonstrated satisfactory performance, with the RF model consistently outperforming the LR model, aligning with previous findings [1, 5]. Despite the substantial increase in features (over 60%) in the custom set, the performance achieved by models trained on both sets was statistically indistinguishable.

Future enhancements could focus on addressing the sensitivity of threshold–based features to erroneous choices. Converting these features to min-max-scaled float values could mitigate this issue and improve compatibility with linear or gradient-descent-based algorithms. Additionally, the custom set includes several features that exhibit linear dependence on others, shown in Figure 2. Dimensionality reduction techniques could be employed to map these linearly dependent features into a new feature, potentially reducing the computational resources required for model training.

References

[1] Kang Leng Chiew et al. “A new hybrid ensemble feature selection framework for machine learning-based phishing detection system”. In: *Information Sciences* 484 (2019), pp. 153–166. ISSN: 0020-0255. DOI: 10.1016/j.ins.2019.01.064. URL: https://www.sciencedirect.com/science/article/pii/S0020025519300763.

[2] Lou Jost. “Entropy and diversity”. In: *Oikos* 113.2 (2006), pp. 363–375. DOI: https://doi.org/10.1111/j.2006.0030-1299.14714.x. URL: https://nsojournals.onlinelibrary.wiley.com/doi/abs/10.1111/j.2006.0030-1299.14714.x.

[3] Alexander Jung. *Machine Learning: The Basics*. Springer, Singapore, 2022. URL: https://alexjungaalto.github.io/MLBasicsBook.pdf.

[4] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. “Estimating mutual information”. In: *Physical Review E* 69.6 (June 2004). ISSN: 1550-2376. DOI: 10.1103/physreve.69.066138. URL: http://dx.doi.org/10.1103/PhysRevE.69.066138.

[5] Ozgur Koray Sahingoz et al. “Machine learning based phishing detection from URLs”. In: *Expert Systems with Applications* 117 (2019), pp. 345–357. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2018.09.029. URL: https://www.sciencedirect.com/science/article/pii/S0957417418306067.

[6] Scikit-Learn. *Random Forest Classifier*. URL: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html (visited on 10/02/2024).

[7] Choon Lin Tan. *Phishing Dataset for Machine Learning: Feature Evaluation*. Accessed: 2024-09-20. 2018. URL: https://data.mendeley.com/datasets/h3cgnj8hft/1.

[8] G. V. Trunk. “A Problem of Dimensionality: A Simple Example”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-1.3 (1979), pp. 306–307. DOI: 10.1109/TPAMI.1979.4766926.

Appendices

	Feature	Value Type
1	NumDots	Int
2	SubdomainLevel	Int
3	PathLevel	Int
4	UrlLength	Int
5	NumDash	Int
6	NumDashInHostName	Int
7	AtSymbol	Int
...	...	...
47	ExtMetaScriptLinkRT	Int
48	PctExtNullSelfRedirectHyperlinksRT	Int

Table 5: Table of all features in the dataset.

A Code

All code I used for this project is publicly accessible at `https://github.com/ancuongnguyen07/CS-C320-ML/blob/master/code/notebook.ipynb`

B Tables of features

	Feature	Description	Value Type	Baseline
1	Frequent-Domain-NameMis-match	If the most frequent domain name in HTML source code does not match the URL domain name.	Binary (0/1)	Yes
2	PathLevel	The depth of the path in URL.	Int	No
3	Insecure-Forms	If the form action attribute contains a URL without HTTPS protocol	Binary (1/0)	No
4	UrlLength	The total characters in the URL.	Int	No

5	PctExtNullSelfRedirectHyperlinksRT	The percentage of hyperlinks in HTML source code that uses different domain names, starts with "#", or using "JavaScript ::void(0)". • request URL% $<$ 20% $\rightarrow$ -1: low risk • 20% $\leq$ request URL% $<$ 50% $\rightarrow$ 0: suspecious • otherwise $\rightarrow$ 1: high risk	Categorical	Yes
6	NumDash	The number of "-" in URL.	Int	Yes
7	QueryLength	The total characters in query part of URL.	Int	No
8	SubmitInfoToEmail	If HTML source code contains the HTML "mailto" function.	Binary (0/1)	Yes
9	NumNumericChars	The number of numeric characters in URL.	Int	Yes
10	NumDots	The number of dots in URL	Int	No
11	PathLength	The total characters in path of URL.	Int	No
12	PctExtResourceUrlsRT	The percentage of external resource URLs in HTML source code. Applied threshold rules. • request URL% $<$ 20% $\rightarrow$ -1: Legitimate • 20% $\leq$ request URL% $<$ 50% $\rightarrow$ 0: Suspecious • otherwise $\rightarrow$ 1: Phishy	Categorical	Yes
13	PctNullSelfRedirectHyperlinks	The percentage of hyperlinks fields containing empty value, self-redirect value such as "#", URL of current webpage, or some abnormal value such as "file://E:/"	Float	Yes

14	NumSensitiveWords	The number of sensitive words (i.e., “secure”, “account”, “webscr”, “login”, “ebayisapi”, “signin”, “banking”, “confirm”) in webpage URL	Int	Yes
15	PctExtHyperlinks	The percentage of external hyperlinks in webpage HTML source code	Float	Yes
16	ExtMetaScriptLinkRT	The percentage of meta, script and link tags containing external URL in the attributes. Applied threshold rules. • external URL% $<$ 20% $\rightarrow$ -1: Legitimate • 20% $\leq$ external URL% $<$ 50% $\rightarrow$ 0: Suspecious • otherwise $\rightarrow$ 1: Phishy	Categorical	Yes

Table 6: Feature Description and Value Types.

C Feature Statistic

D Confusion Matrices

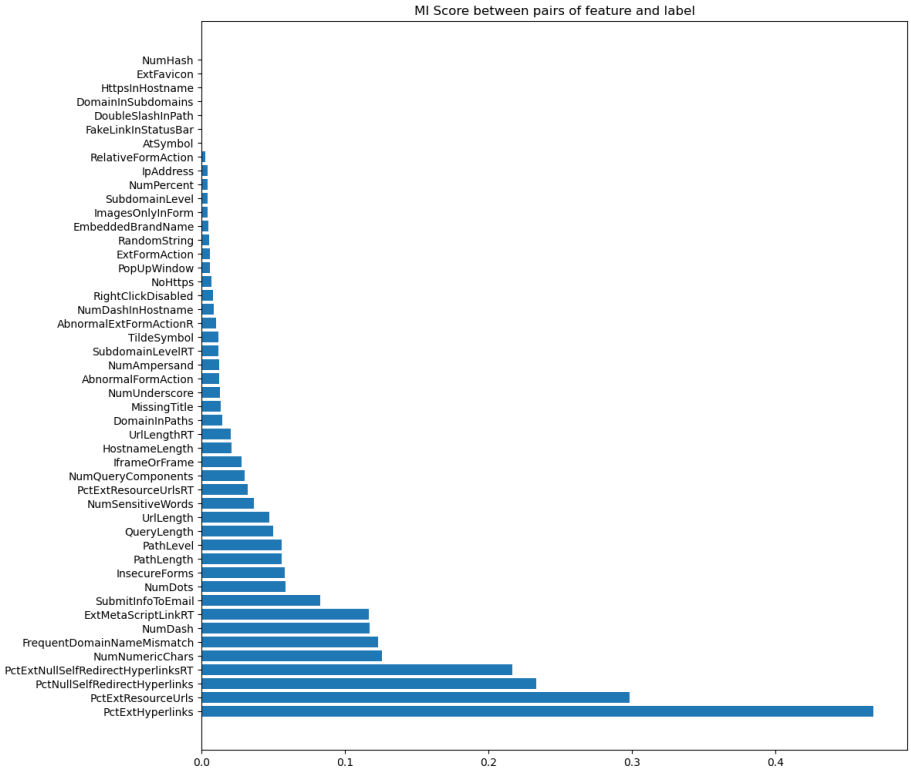

Figure 1: Mutual Information (MI) Scores of all features

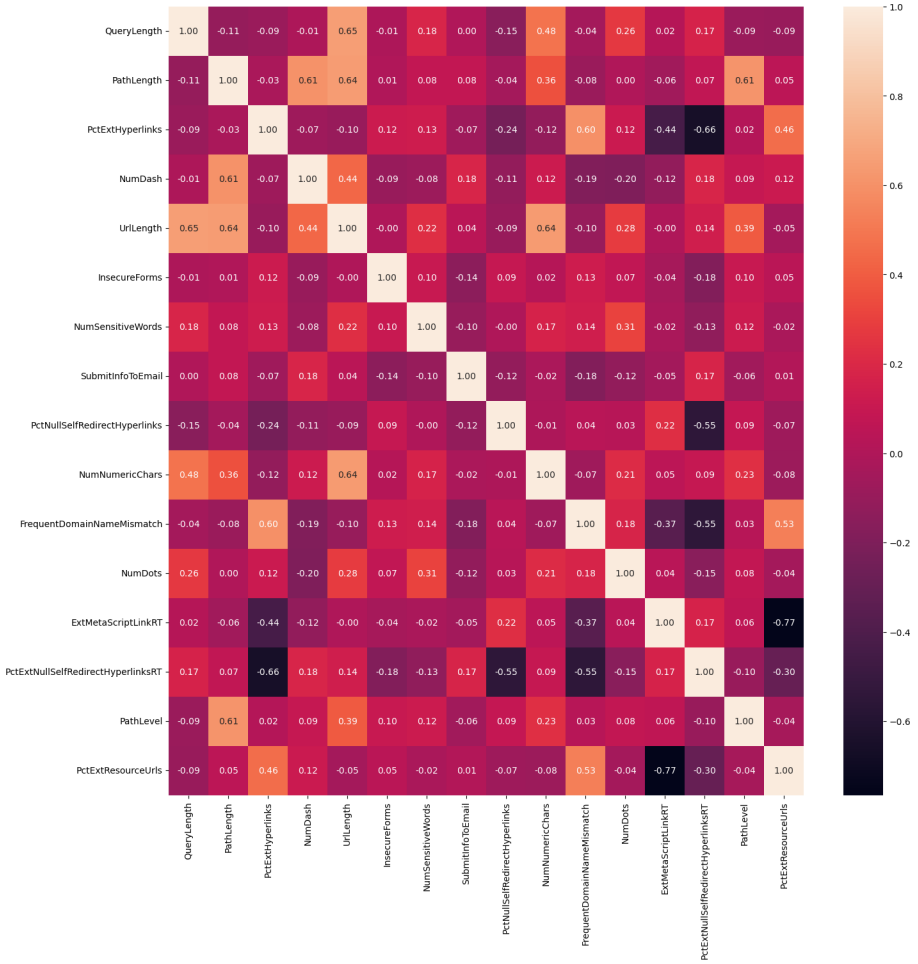

Figure 2: Pairwise correlation between features in the custom set.

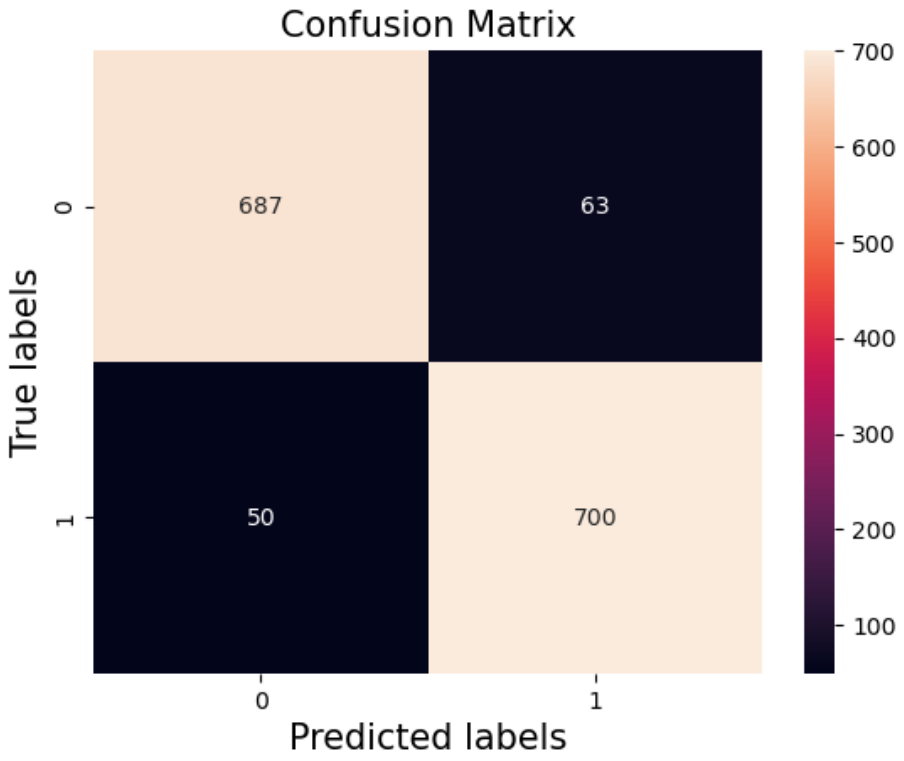

Figure 3: Confusion Matrix generated by the LR model on the validation set.

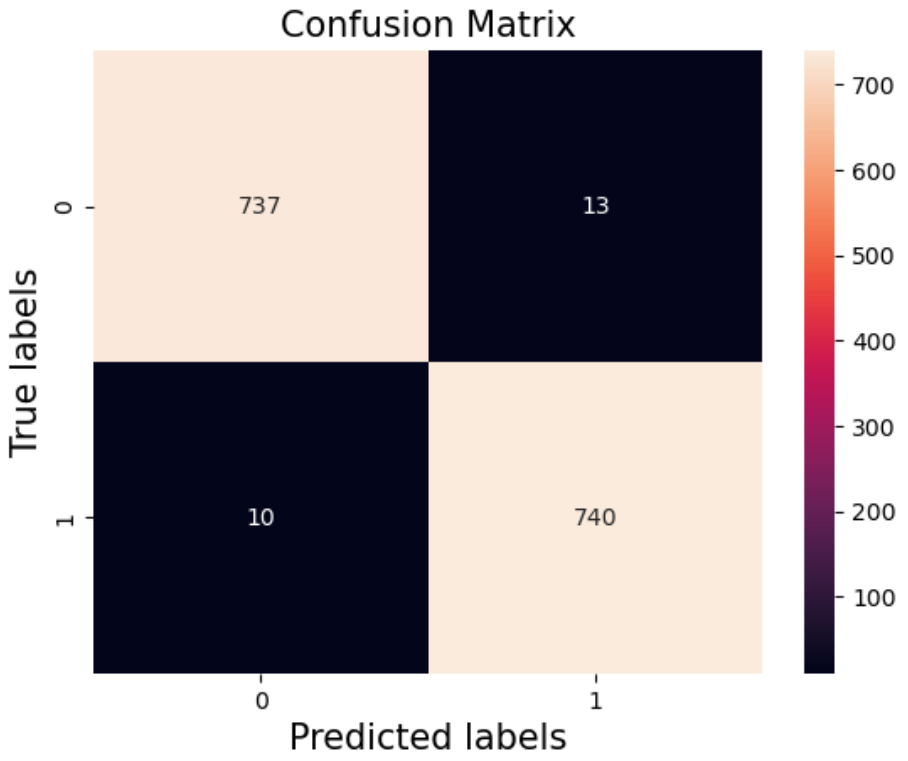

Figure 4: Confusion Matrix generated by the RF model on the validation set.

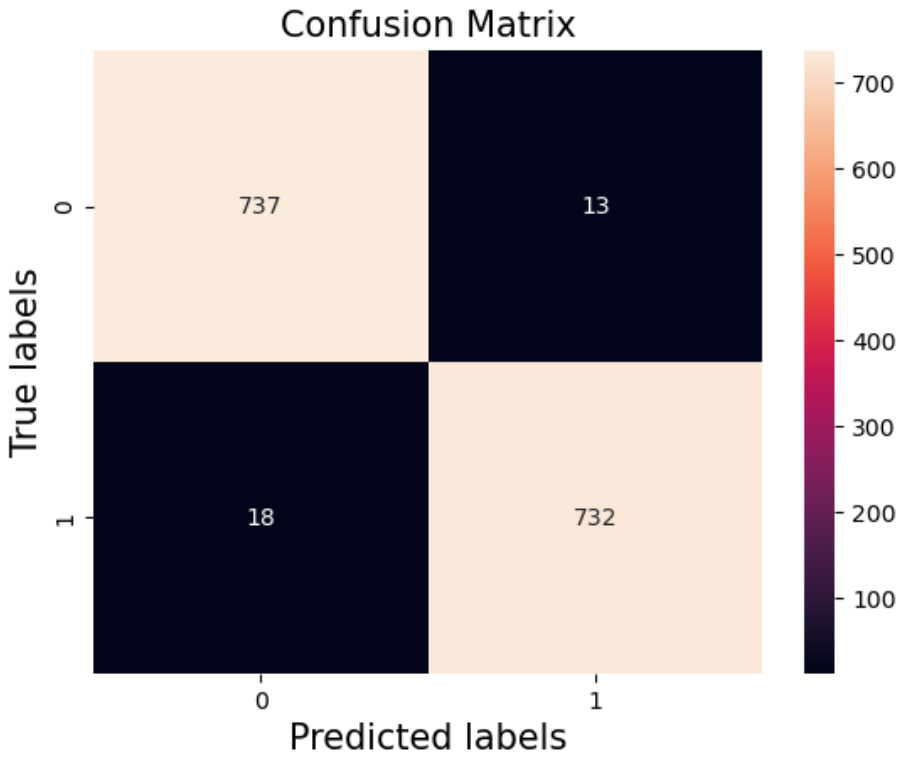

Figure 5: Confusion Matrix generated by the RF model on the test set.

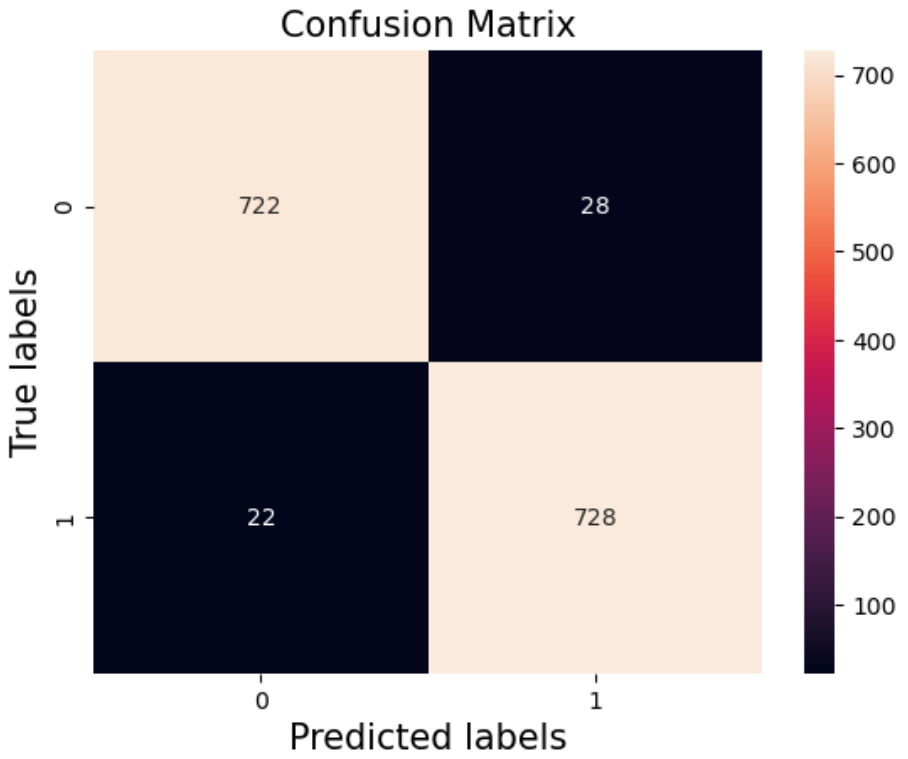

Figure 6: Confusion Matrix generated by the RF model (baseline features) on the test set.