Kaunas University of Technology

Faculty of Mathematics and Natural Sciences

Machine Learning Applications for Detection of Malicious Email Addresses and Fake Usernames

Master's thesis

Edvin Saveljev

Author

Prof. Dr. Jurgita Bruneckienė

Supervisor

Prof. Dr. Robertas Alzbutas

Supervisor

KAUNAS, 2021

Kaunas University of Technology

Faculty of Mathematics and Natural Sciences

Machine Learning Applications for Detection of Malicious Email Addresses and Fake Usernames

Master's thesis

Big Data in Business Analytics (6213AX001)

Edvin Saveljev
Author

Prof. Dr. Jurgita Bruneckienė
Supervisor

Prof. Dr. Robertas Alzbutas
Supervisor

Assoc. Prof. Dr. Valentas Gružauskas
Reviewer

Assoc. Prof. Dr. Tomas Iešmantas
Reviewer

KAUNAS, 2021

Kaunas University of Technology

Faculty of Mathematics and Natural Sciences

Edvin Saveljev

Machine Learning Applications for Detection of Malicious Email Addresses and Fake Usernames

Declaration of Academic Integrity

I confirm the following:

1. I have prepared the final degree project independently and honestly without any violations of the copyrights or other rights of others, following the provisions of the Law on Copyrights and Related Rights of the Republic of Lithuania, the Regulations on the Management and Transfer of Intellectual Property of Kaunas University of Technology (hereinafter – University) and the ethical requirements stipulated by the Code of Academic Ethics of the University;

2. All the data and research results provided in the final degree project are correct and obtained legally; none of the parts of this project are plagiarised from any printed or electronic sources; all the quotations and references provided in the text of the final degree project are indicated in the list of references;

3. I have not paid anyone any monetary funds for the final degree project or the parts thereof unless required by the law;

4. I understand that in the case of any discovery of the fact of dishonesty or violation of any rights of others, the academic penalties will be imposed on me under the procedure applied at the University; I will be expelled from the University and my final degree project can be submitted to the Office of the Ombudsperson for Academic Ethics and Procedures in the examination of a possible violation of academic ethics.

Edvin Saveljev

Confirmed electronically

Edvin Saveljev. Mašininio mokymosi taikymai kenkėjiškų elektroninių pašto adresų ir netikrų vartotojų vardų aptikimui. Magistro studijų baigiamasis projektas / vadovai: Prof. Dr. Jurgita Bruneckienė, Prof. Dr. Robertas Alzbutas; Kauno technologijos universitetas, Matematikos ir gamtos mokslų fakultetas.

Studijų kryptis ir sritis (studijų krypčių grupė): Taikomoji matematika.

Reikšminiai žodžiai: mašininis mokymasis, požymių inžinerija, duomenų tyryba, kenkėjiškų vartotojų kontrolė, elektroninio pašto adresas, vartotojo vardas.

Kaunas, 2021. 83 p.

Santrauka

Kenkėjiškų ar nekorektiškų vartotojų atpažinimas (angl. *detection*) skaitmeniniame pasaulyje yra itin aktuali ir pakankamai nauja tema, kadangi visuotinis skaitmenizavimas skatina ne tik vartotojų, transakcijų bet ir nusikalstamos veiklos perėjimą iš fizinio į skaitmeninį formatą. Naujai besikuriančioje skaitmeninėje ekosistemoje itin svarbų vaidmenį atlieka elektroninio pašto adresai bei vartotojų vardai, kadangi būtent šie asmeniniai duomenys taikomi vartotojų identifikacijai ir autentifikacijai. Esant tokiomis tendencijomis, nagrinėtas poreikis kaip galima anksčiau surasti naujus informacijos šaltinius ir išskirti nekorektiškus vartotojus, siekiant minimizuoti kenkėjišką veiklą.

Šiame darbe atlikta mokslinės literatūros analizė, identifikuotos kenkėjiško vartotojo charakteristikos bei atlikta požymių inžinerija remiantis šiomis charakteristikomis. Tyrimui atlikti surinkti du duomenų rinkiniai, atliepiantys skirtingus pobūdžio elektroninio pašto adresus ir vartotojų vardus (angl. *username*). Toliau, siekiant identifikuoti kenkėjiškus vartotojus, buvo atliktas mašininis mokymasis, naudojant klasifikavimo algoritmus. Kiekvienai išskirtai požymių grupei sukurti atskiri modeliai ir įvertintas požymių svarbumas. Taip pat sukurti klasifikavimo modeliai, naudojantys požymiu grupes ir visus požymius. Rezultatai atskleidė, kad elektroninio pašto adresų klasifikavimas gali būti tikslesnis, nei vartotojų vardų klasifikavimas. Nustatytus klasifikavimo rezultatus galima pritaikyti, kaip įvestį kitiems kenkėjiškų vartotojų pažinimo metodams, siekiant padidinti atpažinimo tikslumą.

Saveljev Edvin. Machine Learning Applications for Detection of Malicious Email Addresses and Fake Usernames. Master's Final Degree Project / supervisors: Prof. Dr. Jurgita Bruneckienė, Prof. Dr. Robertas Alzbutas; Faculty of Mathematics and Natural Sciences, Kaunas University of Technology.

Study field and area (study field group): Applied Mathematics.

Keywords: machine learning, feature engineering, data mining, fraud prevention, email address, username.

Kaunas, 2021. 83 p.

Summary

Malicious user identification in the digital world is a particularly relevant topic, as global digitization is driving not only the transition of consumers, transactions but also criminal activities from physical to digital environment. E-mails and usernames play a key role in the emerging digital ecosystem, as these are the aspects that are used for user identification and personalization. Based on these trends, the need to find new sources of information about the user to minimize malicious activity has been identified.

In this work, the analysis of scientific literature was performed, the characteristics of the malicious user were identified, and feature engineering was performed based on these characteristics. Two data sets were collected for the study, corresponding to e-mails and usernames. Next, machine training using classification algorithms was performed to identify malicious users. Separate models were developed for each selected set of features to assess their significance. Classification models using all attributes have also been developed. The results revealed that the classification of emails was more accurate than the classification of usernames. The obtained classification results can be applied as input to other methods of malicious user detection to increase the accuracy of identification.

Table of content

List of tables

List of figures

Introduction

Global digitalization helps companies to develop businesses, find a new target audience or survive during a crisis when traditional channels are not accessible. Digitalization progress is happening naturally because of the technological advances and digital userbase growth, as well thanks to stimulus from external factors, such as policies and strategies (Green deal (1)) or worldwide pandemic (COVID-19). However, digitalization does not have only positive tendencies, which help companies, society to improve and grow. Widespread digitalization brings new threats and challenges. One of the major threats for companies participating in the digital market and providing services – payment fraud, as this involves direct financial losses and lost revenue if fraud detection is inaccurate. Another vector of digital threats is informational, which is happening in social networks and is usually described as "fake news", "troll fabrics". Both issues are connected to the inability of precise user identification and ease of swapping identities in the digital world.

Fraud prevention using Machine Learning algorithms is gathering interest worldwide, companies providing 3rd-party services to help fight fraud are getting a lot of investments (alone in 2021 mid-April/mid-May received over $240M (2)). One of the underrated information sources during fraud prevention and monitoring – user identifier, while most commonly used is email and username. It can be used in most of the detection and monitoring concepts as additional input for user evaluation.

This research aim is to apply machine learning for the detection of fake usernames and malicious email addresses and create user classifying models based on usernames and emails.

To achieve it, the following tasks were set up:

1. Conduct a study of the current digital market state, identify problematic areas, and review existing malicious activity prevention.
2. Review possible characteristics of the malicious user and develop a methodology for feature engineering and extraction of valuable information from the short text strings (email addresses or usernames).
3. Perform feature engineering and exploratory data analysis of generated features.
4. Train classification models with developed feature sets and prepare final models for each dataset.
5. Classification model evaluation and discussion.

Part of this research was presented during The Students' Conference "Mathematics and Natural Sciences: Theory and Application", organized by the Faculty of Mathematics and Natural Sciences, Kaunas University of Technology, and students' union FUMSA, 2021.

1. Literature review.

In this part of the thesis background of the problem is introduced. For this, a review of the current state of e-commerce and social networks is performed, during which we discuss digitalization (in 2020 EU Industrial strategy it is a key strategic objective). As every service needs the possibility to identify the user and provide him oriented service, the analysis of current identification techniques is performed and a discussion of the pros/cons of those techniques is made. As this thesis is about malicious user identification, malicious activities both in e-commerce and social networks are defined, different fraud prevention techniques are distinguished. Finally, currently used machine learning concepts are reviewed.

1.1. E-commerce and virtual social network growth.

In the last 10 years, the growth of e-commerce represents a phenomenon, which is hard to ignore. Companies, such as Amazon, Booking.com, eBay, Etsy are positioned as "Internet Marketing and Retail" are part of the SP500 stock market index, which covers around 80 percent of the American equity capitalization (3). Each of the top 5 equities by the component weight from the SP500 index has operating e-commerce solutions and sells services in a Card-not-present environment. The creation of such products usually involves big tech teams, which are not accessible for small retail companies, however, small companies can use e-commerce solutions provided by other tech companies. In 2017, WordPress's open-source content management system was used by 30% of the websites and the WooCommerce plugin, which adds e-commerce features, was the most popular plugin for the content management platform (4). Mentioned software solutions help companies to create e-shops without large investments, thus small businesses are motivated to open more online stores and that is why each year it becomes nearly impossible to forecast the growth of e-commerce. For example, in 2017 Statista.com tried to predict the growth of the e-commerce market in the United States, with an estimate of 315 billion U.S. dollars of revenue in 2018, however, it appeared to be 1.6 times bigger than predicted, hitting the mark of 2022 from the same forecast (5). Not only technological advances but as well political decisions (for example the Green deal (1), where digital and green changes are expected to be correlated) stimulate the growth of e-commerce and current forecasts are that e-commerce next years will have around 15% Year-over-Year growth (Fig. 1). It is important to mention, that digitalization is one of the key points in the Green Deal and EU Industrial Strategy introduced by the European Commission (1).

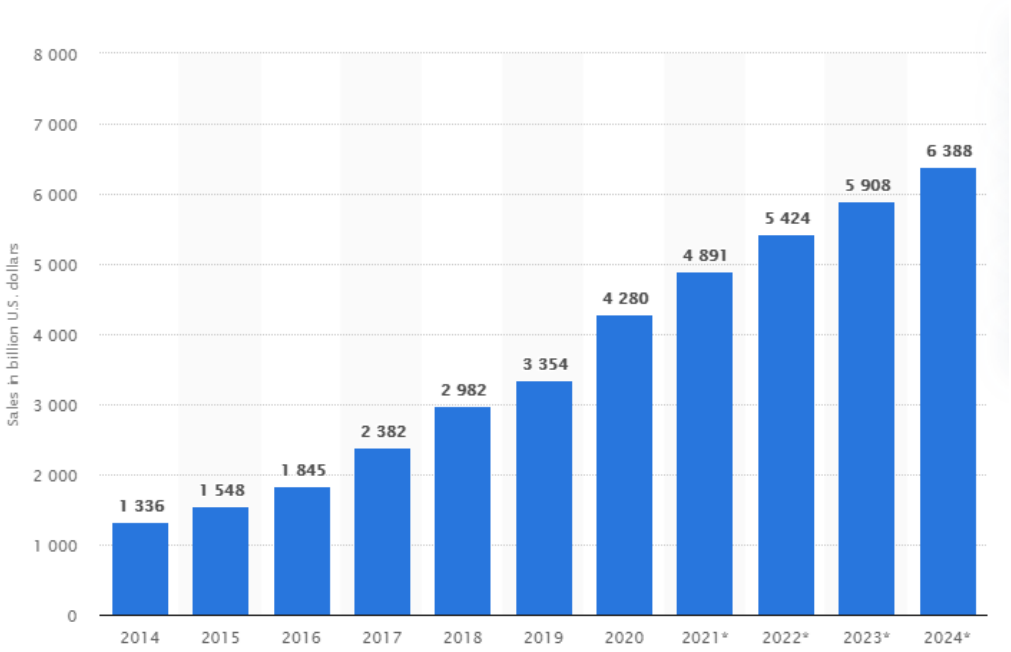

Fig. 1. Retail e-commerce sales, historical data, and forecast 2014-2024 (6)

WooCommerce and other solutions which provide ready solutions to open e-shops such as Shopify, e-Pages, and MyCashFlow receive the increase in demand during the COVID-19 lockdown, as the turnover of e-commerce in some countries increased by 60% or more (7). This turnover dynamic can be explained through a change of corporate politics to decrease the negative effects of a pandemic by adapting business models to the new environment during lockdown (8). Changes can be classified into three main groups (9): development and adoption of alternative channels for service and product delivery, for example, e-commerce solutions; new product line implementation, using existing resources and infrastructure (usually, manufacturing of medical products); focus additional human and technological resources for products which demand increased during pandemic time. Additionally, digitalisation, automation, new technology and innovation implementation help companies to decrease possible loses and continue operations during pandemic (10).

Usage of social networks represents another phenomenon that we observe from the beginning of the 21st century. As of January 2020, there are more than 250 social networks, which can be classified by different functionality and target audience (11). Research made by Hootsuite suggests, that in developing countries most of the population maintains a presence in virtual social networks thanks to developed communication technologies (12). By the beginning of 2020 mentioned 250 social networks had more than 3.8 billion active users (11), Facebook being one of the biggest virtual social networks, has monthly more than 2.8 billion active users (Figure 2), while 1.84 billion individuals visit the website daily (13). While most of the social networks try to move in one direction, providing several functionalities, Facebook tries to mimic competitors – copying stories from Snapchat/Instagram, video collections from YouTube, Marketplace, and even dating option, to retain the user in its ecosystem as much as possible.

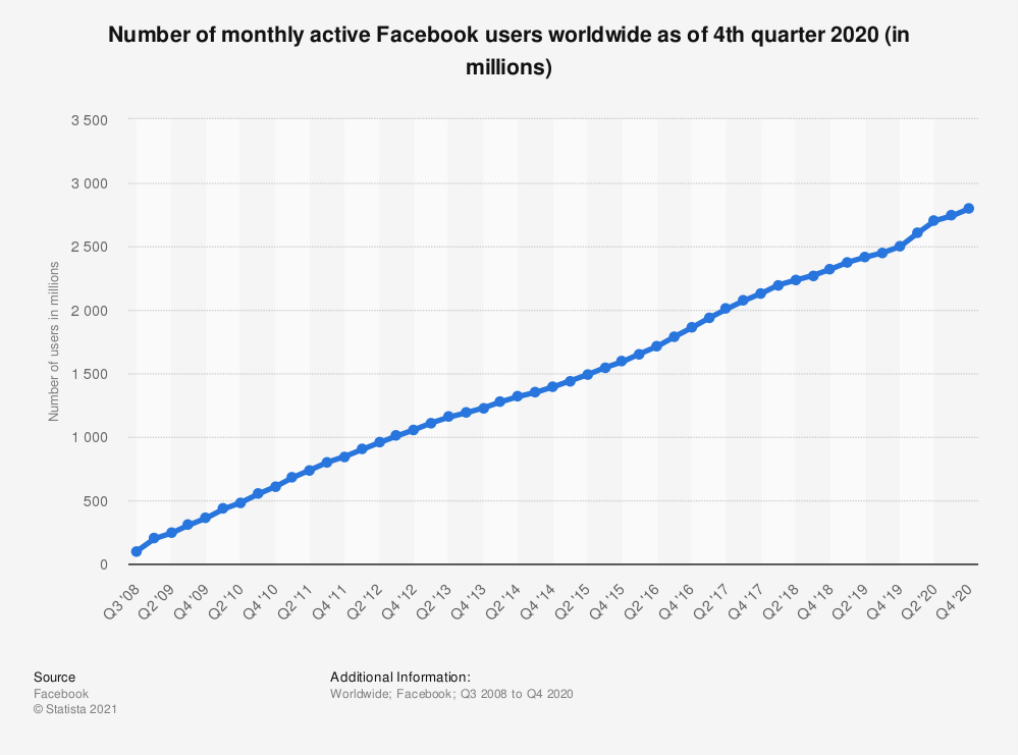

Fig. 2. Monthly active users in one of the largest social networks – Facebook.com (14)

1.2. User identification in web services.

Market hyper-growth and constantly growing user base for services and social networks create problems of user identification for web stores or services. Historically there were different approaches to user identification, starting with personally assigned user id, which was not successful, as for each service – the user must store somewhere not only the password but username as well. Governmental portals and banking institutions tend to use personal identification numbers, social service numbers or give specific user-id after identity confirmation. Although this idea of identification seems appealing, as the service would be sure, that this is a unique person and details of the user are correct, however for person identification sophisticated confirmation is needed, where access to governmental databases is required. Such access might give private companies the ability to abuse confidential information gathering, which may increase the chances of sensitive data leaks. Because of the increased threat of abuses or data leaks sensitive data providers normally create compliances and perform constant audits (15). For private companies, especially for smaller one's requirement implementation seem hard to achieve, both from a technical side and is financially unsustainable in a long run especially if identity verification is not constantly needed (16),(17). Additionally, applications, where such identifications can be performed must be secured and constantly updated, researching possible vulnerabilities of communication protocol (18), to avoid data leaks from the user side as on average each data leak cost in fines and public image is around $6,5 million according to survey made in 2015 (17).

At the end of the 90s, ideas of email-based identification and authentication saw a raise, competing with the mentioned traditional public-key identification and other potential options, such as postal mail or phone verification (19). This identification type may seem less secure, however, for service providers, such identification type removes the necessity of implementing compliances, as user

identification may be made by sending a confirmation link or one-time password. All initial security risks are manageable in a conditional manner, as most services after email verification provide only basic functionality. For example, private companies, which provide services to other companies, use email identification for regular functionality, requiring users to contact supervisors or managers to change sensitive settings. Such identification partly removes the liability of constant data protection from the service provider side, as this makes users responsible for their data protection. The service provider becomes liable only for the data stored in his databases. The same concepts are applied for e-commerce and social networks in the general case, for basic functionality it is enough to confirm email, however, in case of increased usage – the user is asked for additional inputs (phone, address, bank statement, photo/video confirmation). Such a concept may decrease friction between the user and service, as such verifications are way easier to perform, comparing to verifications with ID or SSN. Additionally, because of the uniqueness of the email address, such verification guarantees, that the user has access to the account, and most email delivery security concerns are solved nowadays (20).

Although email verification solves verification issues from the user side, it has its disadvantages. With the increase of free email service providers, it is harder for companies to prevent redundant registrations from the same users and stop potential abuse from the user side (21). This includes abuse of loyalty programs, performing suspicious activity on one account, and switching to a new account with fresh history to perform the same suspicious activity again. From a user perspective such practice has its disadvantages as well – losing access to the mailbox leads to losing access to all accounts, additionally, in the case of password leak – all associated with email accounts are in danger. Moreover, to decrease chances of certain people's email addresses being targeted by social engineering or vulnerability searches (22) - social networks and other services, where certain user profile info can be accessed – nicknames and usernames were introduced (23). One of such examples is a Twitter username and this gives users a sense of anonymity, which sometimes makes services using usernames a toxic environment, where users blame each other and post abusive messages in response to the message of other users.

1.3. Device and user fingerprinting in web services.

In the past apart from email verification, it was possible to get a lot of additional data points about devices through browsing applications, intercepting, and analyzing network data between the user and server (24). Having extra data about the user would lead to possibilities to cross-reference data and create fingerprints and identify the user, even when he is not logged in based on his device. However, because of the privacy concerns in the last 10 years, most of the vulnerabilities were patched down so the accuracy of such fingerprinting decreased dramatically. Moreover, internet users are switching to devices in which hardware is predefined and customization is quite complicated, comparing to how it was 10 years ago, where everyone tried to build custom setup, or there were hundreds of cell phone models. Access to the storage is getting limited, browser User-Agent definition is getting exchanged to client hints. The last change is quite big in privacy and fingerprinting, as previously service was asking the client for information (usually not related to the service but for better fingerprinting), now the client will answer only to certain requests (25). VPN and Proxy servers are getting widespread, so this fakes some information, which could be identified previously from the network packet analysis. This unification and challenges lead to the point, where fingerprinting techniques are either getting way too complicated for robust identification (26) or researchers must find new ways to identify the user. Currently, one of the raising concepts is

behavioral fingerprinting, where primary concepts were described a long time ago, such as keyboard stroke, mouse movements, reading speed. However, because the complicated implementation and behavioral analytics are at their early stages, not many breakthroughs, comparing to hardware and device setting fingerprinting. Some commercial products such as FingerPrintJS claim to be able to identify the user with 99% accuracy, however, it is almost impossible to find research, which could confirm such claims at the current stage of behavioral fingerprinting.

1.4. Fraud in a card-not-present environment using fake identities.

The history of fraud in the card-not-present environment starts from the beginning of credit cards. It constantly evolves with the changes of technologies and security protocols, gradually disappearing from real-world applications into the virtual environment. At present, it is practically impossible to successfully perform so-called strong authentication over the phone or other means, as most of the issuers accept transactions only with EMV chip scanned and PIN verification. However, in the virtual environment, on the internet of things, there is still a lot of ways to perform fraudulent transactions. That is why each year, numerous organizations publish reports about raising threats and trends in fraudulent transactions. For example, in research made in 2018 (27), it was calculated, that overall turnover in 2017 with the UK issued credit cards in the card-non-present environment was a total of 154 billion pounds, whereas fraud with the UK issued cards in card-non-present was accounting for 70% of total credit card fraud for 432.3 million pounds. The calculated fraud rate from the provided numbers is around 0.2% which is lower than the fraud rate defined by card schemes (28), however, industries, where digital goods or subscriptions are sold, have higher rates. On the territory of the Single European Payments Area (SEPA), it was reported by European Central Bank, that card, not present payment fraud takes 79% of total payment fraud with the highest number in France (29). Another research made by Nilson Report, suggests that Worldwide fraud losses exceeded 30 billion dollars and it is expected, that by 2027 those losses will reach 40 billion dollars (Figure 4). Global trends during the last years, show, that the volumes are still growing and during the COVID-19 pandemic, although the percentage of fraud between card-not-present transactions decrease (30) this may be not because current counter-measures are super effective, but because the volume of payments increased.

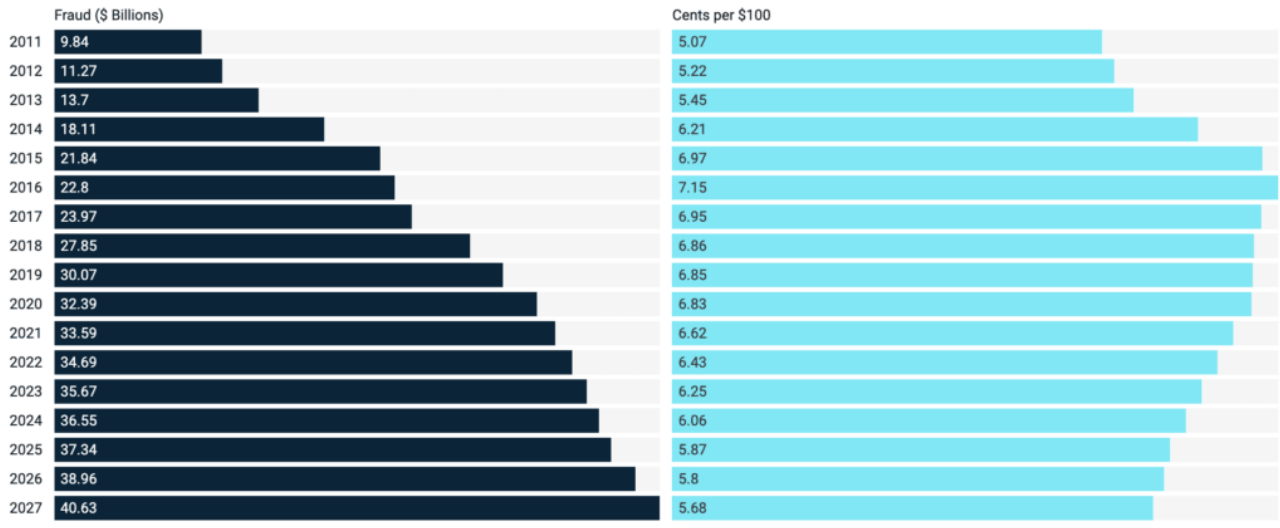

Fig. 3. Payment fraud losses more than tripled, comparing to 2011 (31)

As fraud e-commerce is developing, each year more companies, providing services and selling goods are targeted by fraud. As the research made by one of the largest US banks, JP Morgan states, each year more and more US companies were targeted by payment fraud – in the 2019 year 8 out of 10 companies were targeted by any type of fraud attack (Figure 4).

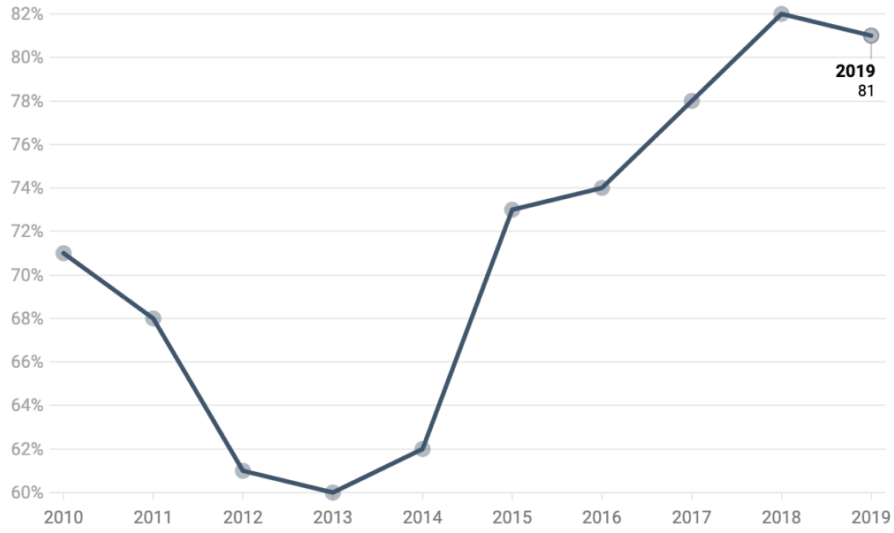

Fig. 4. % of US companies targeted by payment fraud (32)

Although card schemes are inventing new security protocols (so-called 3DS1.0 and now 3DS2.0) and countries are using regulatory power to enforce those protocols (for example EEA region implementing PSD2 compliance) card-not-present fraud is a remaining issue. One of the reasons is that such standards as 3D secure verification have the way of frictionless authentication for the trusted customer, where risk is managed from the issuer side and this can be abused if the fraudulent agent knows how to tamper and what data is needed by the issuer (33). Moreover, as mentioned in the previous topic and the provided research – most of the attributes for holistic determination, if the authentication is needed, are gathered from browser and browser settings. This leads, that tempering the victim's fingerprint does not require sophisticated techniques or malware, it is enough to find out which device model the user has and approximately tune the attacking device settings. In countries where financial regulators do not require the issuers to comply with PSD2 regulations, issuers are choosing to decrease the friction for their clients as much as possible if merchant enables 3DS verification and because of that, they decide that it is easier in some cases to reject the transaction, rather than asking the client for 3DS verification.

Nowadays not only country regulators are attempting to implement strong authentication, but card schemes as well, that is why most of the card schemes already have implemented liability shift mechanics (34). This means that during transaction communication, both merchant and issuer communicate about 3DS authentication and after the transaction is finalized if it is reported as fraudulent, the party which had less version of 3DS or did not ask for its verification will take financial responsibility. Apart from the liability shift stimulus – credit card companies have excessive fraudulent transaction programs, where one of the conditions to exit monitoring programs and avoid fines is implementation on a certain percentage of transactions 3DS (35).

As the identification and verification standards were designed to be secure and at the same time frictionless for good customers, later one advantage became one of its main flaws, as it relies primarily

on fingerprinting, which is getting quite outdated. The existence of such disadvantage neglects advantages that are coming from enhanced security, as in some cases it may add friction for good users while giving no extra friction for bad users. Liability shift protects from getting financial implications, however, fraud levels for merchants grow and they still maintain the possibility to get into excessive fraud programs, where they start getting fined if they do not low the fraud level (28). This fact leads, that merchants must perform investments in transactional security (implementing rule-based engines or ML-based fraud detection services) and R&D departments or use 3rd party solutions, which provide different verification/validation/identification services.

1.5. "Fake news", public opinion manipulation, and other suspicious activities in virtual social networks.

"Fake news" is a term which popularity skyrocketed in 2016 after the election of Donald John Trump as it was one of his most popular accusations towards news media corporations. The meaning of it – false information, provided as a sensation under the guise of news reporting. In 2017 Collins Dictionary had "fake news" in the top-10 shortlist for the word of the 2017 year (36). The same dynamic can be observed via GoogleTrends infographics (Figure 5), as the word was almost not searched until the end of 2016, afterward it is commonly searched with some sparkled interest during the beginning of the COVID-19 worldwide pandemic, when there were accusations by some groups of people towards news media, that the COVID-19 is fake, and lockdown is not necessary. One more of the definitions of "fake news" is provided by The European Commission, which defines it as "intentional disinformation spread via online social platforms, broadcast news media or traditional print." (37). As this problem is raising The European Commission created an institutional office name Disinformation Review, which task is to police media. This office consists of over 400 experts, media representatives, and NGOs, which report disinformation news to EU officials and later to society (37).

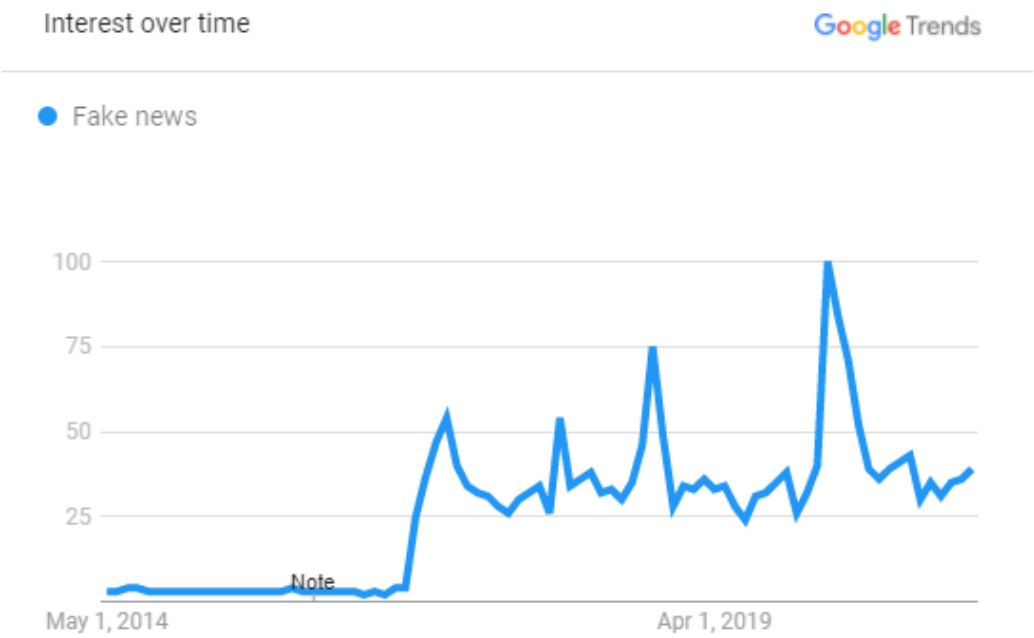

Fig. 5. „Fake News" search term interest over time by GoogleTrends (38)

Opinion manipulation exploiting informational technologies represents a very old phenomena. Historically some countries maintained or have their Ministries of Propaganda, to affect the public opinion of their citizens on certain topics (39). However, as the communication and information gathering transferred from traditional media to virtual media and with the ease of virtual social networks, it became easier to find people sharing common interests new threats appear. While "fake news" operates usually in conventional news media or mimicking them, opinion manipulation is more often associated with social networks. With enough number of automated users, it became obvious, that it is possible to imitate social groups of users and affect real people, which may have an interest in collision with the artificial social group. The first massive public appearance was discovered in 2014 during the Crimea Russian-Ukrainian situation when Russian Company "Internet Research Agency" was employing people for posting messages with agenda in social media such as Twitter/Facebook from synthetic identities and thus form a public opinion (40), (41). Later it was discovered that this organization was already working successfully for some time, operating not only in mentioned networks but also influencing ideas from video platforms like YouTube locally and abroad. The next major scandal involving a similar technique was observed during the election of the USA president in 2016 when numerous fake profiles on Facebook were forming public opinions of presidential candidates, journalists and later Facebook managed to find links to the earlier mentioned Russian organization (41). Same trends were noticed during other events and such techniques can be expected to form public opinion about the entity and make some public companies' profits or losses. During analysis of user-profiles involved in mentioned social attacks, it was observed that all of the users used generated or stolen profile pictures, using fake names, and usernames/nicknames looked generated (42).

Public opinion manipulation is not only a vector of attack using Twitter and Facebook botnets, other activities as malware or spam distribution, social engineering, or raising hate between social groups (43). As data breach and public opinion investigation in the aftermath of Cambridge Analytica scandal – Facebook found that the same agency organized group "Blacktivist" for events supporting the "Black Lives Matter" movement. They managed to get more followers than the original movement group because of the targeted advertising. At some point they intentionally organized two rally meetings in the same place both for "Blacktivist", as well for far-right "Blue Lives Matter" in the same place, to create chaos and raise hate in society (44). Both mentioned social networks are releasing news, that they are fighting bots and such malicious users, providing data, that less than 5% of the tweets are posted by bots, however, some researchers are stating, that the numbers are higher (45).

1.6. Possible malicious user detection techniques

Both mentioned earlier malicious activities – payment fraud and opinion manipulations share the same core issue – the inability to identify or verify the user precisely. In general, there are three different approaches to targeted detection of malicious activity (45):

- **Social-Graph Based Techniques**. In Social-Graph-based methods, user information is stored as documents. In documents normally stored information is separated into two categories – user identifying and transactional. User identifying information in both domains of Social Networks and Payments usually consists of unique identifiers, which could link to the unique user (such as stored cookies, calculated fingerprints, and other identifying information). Additionally, as user identifying parameters different domains can add different data points:

for example, social networks – social connections(46) (followed users, followed topics), payment and e-commerce industry – payment information (tokenized used payment accounts). Apart from user identifying information – transactional information can hold information, which is not static for the user (device model, type, IP, etc.). Based on information stored and common user behavior in such detection methods, the assumption that good and malicious users create different user groups (or clusters/segments). If one of the users in a cluster is malicious, all other users and new users, which get connected to the same cluster and have similar transactional data would be identified as fraudulent. This method has a major drawback, as the user identifiers must be unique and should hold precise information about the user, and finding out such identifiers is not an easy task. Moreover, as the users are classified by identification data, changing identification information would through the user out of the cluster and he would avoid detection.

- **Human-in-the-loop Methods.** Those methods as the naming hints use experts for the final decision, where transactions/users are manually reviewed by experienced analytics. Review procedure can occur on all transactions or only on the selected ones. Selection is usually made by another type of Method. In both cases, this needs a lot of resources from human experts and that is why it is very expensive and not very scalable (47).
- **Feature-based Techniques.** Those techniques are widespread around. The main idea is to gather as much raw data as possible, create features around it, and filter by its transactions/users. Filtering is usually achieved from static thresholds, a complex set of rules, or ML models. Feature engineering is not only interpreted as preparing raw data, but as well another type of technique, as using data from other models, using historical data, or using anomaly detection algorithms. Feature-based techniques usually rely on the amount of raw data that can be received and the quality of the features, which can be calculated by processing raw data. In this research, we will focus more on those techniques, as those solutions are usually very scalable and can be used as additional information for other techniques. Below are use cases of the provided technique.

In the case with the credit card, there are limited ways to identify correctly if the user possesses the credit card (for example, absence of 3DS on the payment card or activation of frictionless identification) as the e-commerce cannot acquire all necessary information about the user from the issuer or verify user's credit card through banking application, due to compliance. During the transaction time, the merchant has only a limited amount of information from the issuer, which is: first 6 and last 4 numbers of credit cards, expiration date typed by the user, and few markers if the issuer performs validation (48). This leads to the merchant's responsibility to mine as much information as is possible about the user on his own and build a possible profile of the user. If the merchant does not have the ML models, most of his rulesets are based on the velocity of parameters. Those measures seem effective for most of the cases, but velocity-based rules have some major flaws: they are very sensitive to attribute change, as changing one, no matter it is a device or payment would reset velocity count. This may lead, that fraudsters will have a transactional window to perform more fraudulent transactions, while regular users would rarely try to change and remain over blocked. In case if the merchant is using ML models, he can create a lot of different feature sets, based on the market, market coverage, and amount of data points are collected (48).

During the identification of bot messages or malicious activity in virtual social networks commonly used detection techniques are text analysis techniques. Such techniques typically try to extract

information and malicious keyword, perform linked content analysis to detect the nature of hyperlinks. On the other hand, text analysis methods usually experience challenges of scalability, topic evolution, or simple symbol exchange (one language symbol changed with similar ASCII symbols not related to original language) (49). Although creating specific data preparation-cleansing pipelines or adapting anomaly detection algorithms may solve part of the issues, however, text analysis tasks are very complicated in general, requiring larger data scientist teams and a loss of time and infrastructure investments, as text analysis is a resource heavys task. Moreover, as sometimes bots are having discusions and communication, having opposite views, this task becomes even more complicated (50).

Typically, one of the attributes, which is provided by the user is underrated – email name or nickname, due to challenges to extract information from a small amount of text. During fraudulent attacks or impersonation (Account Take-Over) fraudulent users tend to create a lot of fake identities by hand or computationally, which leads to a lack of creativity, thus making it easy to detect such email or nickname patterns manually, however without extracting features or receiving email score from 3rd party risk tool providers (for example LexisNexis EmailAge email score) this info for automatic rules is typically not used (45).

1.7. Fraud and anomaly detection using machine learning.

There is currently a broad variety of various methods and techniques to detect fraud or anomaly detection. With the development of algorithms and technology, artificial intelligence becoming the new buzzword – the demand for an excessive number of risk analysts is decreasing. More and more of fraud detection is moving from manual analysis into automatic detection of implementing algorithms, models in transactional and graph databases. Ability to evaluate transactions swiftly not only decreases the cost of each transaction but as well decreases the payment friction, as fewer transactions require manual review. There are numerous advantages of using the ML-based models, as they are less prone to bias, can discover novel undetected combinations of patterns, identifying fraud more accurately with a lesser false positive rate.

Standardly, there are a lot of different ML algorithms which could be tailored for solving fraud and anomaly detection, but most of them can be defined by the concept in three main groups (51):

1. **Unsupervised clustering.** This type of concept assumes, that there is no prior information about the classes in provided data, the data is static and anomalies along with malicious users are distanced from the good transactions and can be grouped as fraudulent transactions. One of the biggest disadvantages of this method is that for the model to be trained, the training algorithm must receive as much as possible (usually full dataset) about the transactions. This led to large training costs, another big disadvantage is the assumption, that transactional traffic will not change, which means, that there are now significant traffic shifts and the new data constantly follows old patterns. However, in real-life applications, for most of the problems, such methods are quite unrealistic, as the data changes constantly, especially from the start of prevention.
2. **Supervised classification.** During this training method, the algorithm trains on the data to detect both good and bad users, training is performed on the labeled data. Those methods are prone to traffic shifts, the same way as unsupervised clustering suffers and are quite sensitive to data mislabeling. In the cases where training data is correctly labeled, there is still a need for as little traffic shift as possible. Another issue using algorithms sharing this concept is that

fraud and malicious activity usually are observed in small quantities, therefore there is a need for oversampling usage and other techniques. Most of those imbalanced data detection techniques usually help during modeling, however, it increases bias and does not work great in a production environment.

3. **Semi-supervised.** Usually, the motivation behind this method is that the ML algorithm observes only good transactions, and there is the inability to fix this instantly. Although this concept sounds appealing, however, this methodology requires data to fulfill both assumptions to be correct. Mislabeled transactions and constantly skewed transactions harden the prevention of such transactions.

Most used techniques in real-time fraud detection are supervised and semi-supervised concepts, as they do not require as much computational power, to create or retrain models. However, unsupervised clustering may help to improve model results, as the transaction can be segmented, and each segment may have different models. However, the same effect can be achieved with model assemblies using supervised and semi-supervised training routines.

1.8. First section summary.

Digitalization of businesses and switch from "the offline" world into the "online" world is happening and is hard to deny it. In most cases, it is quite possible to call such growth hypergrowth and there are several reasons for it – technological, political and social. As well, the COVID19 pandemic had its influence, as a lot of businesses and applications had to change production vectors or scale up in a virtual environment to decrease possible losses. During the pandemic, the popularity of social networks increased as well, as people were trying to connect. However, internet services have their threats, some of which transferred from the real world into the virtual, a few of those are – payment fraud, opinion manipulation, and fake news. For those threats switch to online in some cases means more possibilities, as identity verification in an online environment is easier compared to regular peer-to-peer identification. Merchants and other services are forced to create or use 3rd party services to decrease possible fraud and sometimes are too harsh for their users, as prevention techniques are not ideal and regular user experience is hurt. The vast majority of the services in the digital world use an email address as identification for their users. During traffic filtering with automated methods, email and username features are rarely used, as it is a rather hard task to extract features from a small amount of information.

As a result – the proposed solution is developing email evaluation models to help to filter possible malicious users. The calculated score can be used as the additional feature/input for other detection methods.

2. Methods and concepts used in research.

The following part of the thesis describes the source of data gathered for the research project, as well the review of various methods and techniques used to extract information from the raw data later used during the training process of fraud detection. As it is proposed in the summary of the literature review, the main goal is the development of an identifier evaluation score, which could suggest the probability of the username or email address belong to the user with malicious intent.

2.1. Problem domain and data collection

As mentioned in the discussion, the domains for which build-up solutions are proposed are – social networks, service providers, and e-commerce providers. During this project, the goal is to build machine learning-based models for those domains to build up scores, which would represent the probability of the email or username being fraudulent (abusive).

2.1.1. Good email and username list sources

For conducting the research, several data sources were connected for building up positive classes of data sources. As this research requires lists of emails, which belong to good users for purposes of research – data breach leaks were used. Regularly those breaches consist of personal information of users, user identifiers, sometimes encoded passwords, contact details. In some cases, the breach consists of usernames or payment information. As this research is focused on username and email verification, only datasets containing at least one of such identifiers were used.

After the good email dataset is created, all the information apart from email or usernames was erased, as this information is not relatable to the research. Although in future research geo-information of the user could be used to achieve better results.

For usernames, as it is hard to find usernames in email leaks, datasets involving Twitter messages were used. All data, apart from the username after the user is identified as the good one is erased, as that data is not relevant to this research.

2.1.2. Fraudulent email/username list

For a collection of email and usernames, several types of data sources were used – ML learning datasets, where fraudulent emails were marked as fraudulent, datasets from Reddit, where users shared emails of possibly fraudulent abusers of programs. The additional source for datasets is the CleanTalk.org website, where partnered forums and websites share spammer emails, for the shared database, to help websites prevent more spam.

As for the usernames, the biggest source of fraudulent usernames came from Twitter datasets, where it is implied that users, who participate in malicious activity are those users who have a low number of followers and spam the same message as other users having similar characteristics.

Both datasets have all information apart from usernames and emails removed, as this project is targeted only to evaluate the username/email based on its build-up.

2.2. Data preparation

After preparing datasets, they are stored in csv or txt format, where a single email represents the row. As those emails are the only raw data we will be using, there is not much that we can do in the first steps, however, the email structure is described by RFC 5322 (52) and updated in 2013 by RFC 6854 standard (53).

According to the standard emails have local-part, divider "@" and domain. The domain can be represented as the domain name or IP address in the bracket, however, the second option is not widespread, as the domain can be hosted on the same servers and share the same server IP, thus making mapping and mail forwarding extensively hard task for system engineers.

There are different requirements for the local part of the email, however, there exist several common requirements which should be overseen during the preparation of data, and during feature generation:

- Can be both quoted in quotation marks and unquoted. If the local part is quoted, more symbols are allowed, however, most of the email servers try to avoid allowing this functionality, and indeed RFC 5321 standard (54) warns server owners to avoid allowing quoted string as local parts of emails.
- The local part is case sensitive, but the same RFC 5321 standard (54) suggests, that email service host, should not define mailboxes using different cases as unique emails, instead, it is a better practice to ignore cases. This property can be used as part of the feature generation, as good users may try to separate names/words with different cases.
- The possibility of adding tags to the local part, means, that email may contain a tag, which will help the user to identify mails. Possible ways – adding a tag in brackets in the beginning or end of the local part, or provider-specific methods, most of the email providers support wildcard recognition of the email, where the local part is everything before plus sign and tag is after the plus sign. So for an example: jonas.jonaitis+tag@example.com/ (tag)jonas.jonaitis@example.com/jonas.jonaitis(tag)@example.com could be delivered to the same mailbox, defining in the header of the email address to which it was sent. However, most of the email services allow only tagging with pluses.
- Dot symbol usage is allowed in case it is not at the beginning or end of the string, and it is not consequently repeating. Some email providers such as Gmail.com ignore dots during the delivery of the email. This way user can tag their emails or use dots to separate unique parts of the email local part.
- Permitted characters in the local part are only ASCII: Latin a-z, A-Z, digits 0-9, and printable characters - !#$%&'*+-/=?^_`{|}~.

According to those requirements, there is a need to check if the collected data correspond to those requirements from standards and as well, we can already observe, that some of the standard properties can be used as features. Depending on the balance and issue – different approaches will be used: entry removal or data imputation.

For usernames in Twitter, Twitter has the next specifications, which allow users to use alphanumeric characters (A-Z, numbers 0-9) and underscores. Username is not case sensitive, and the length of the username should not be longer than 15 characters and not shorter than 4. Unless the username is official, it is impossible to create a username, which includes Twitter or Admin strings. An additional recommendation from Twitter is to use shorter names, as they represent better people.

According to the Twitter requirements, collected data will be checked if it follows all the raised requirements by documentation.

2.3. Methods and concepts used in feature engineering.

After the data preparation step, when it is validated, that all entries follow the requirements made both by standards of Twitter and RFC, which regulates possible email structures feature engineering can be started. After reading the papers from the forensics field and spamming prevention, users performing malicious activity try to maintain such characteristics (55), (56), (57):

- Malicious users try to stay anonymous by generating complex and diverse information.
- Malicious users are demographically biased, in the context of the email and username verification, this will be mean more, that malicious users are acting as anomalies comparing to the good user distribution. However, it is very hard to determine the user's demographical properties from a minimal amount of text.
- Malicious users are efficient. As they have limited time to perform such attacks – the majority of such users use principles of quantity over quality. This way malicious users try to spend their time efficiently, by optimizing typing behavior and avoiding using convenience functionality provided by RFC standards.
- Malicious users are similar between themselves and different from good users, this means, that users tend to have similar characteristics mentioned, and when they generate new usernames/emails they follow the same patterns. This statement is motivation, that ML models, based on such features could be working.

Based on the mentioned characteristics in the following subsections, we will review possible calculation techniques and used the information to create feature sets.

2.3.1. Complexity and information diversity

There are different approaches for complexity measurement:

- Algorithmic Information Content and Logical Depth, where complexity highest values are observed within a random process.
- Statistical complexity and Physical complexity, where on the opposite, complexity is the opposite of randomness.

Algorithmic Information Content was offered by Kolmogorov as the amount of information, which is required for the optimal information encoding (58). According to the algorithm proposed to the Kolmogorov and later used in information compression, frequently repeated characters or strings can be encoded more efficiently, thus making unique strings words have bigger complexity measurements. Opposingly – not repeating characters in the string make the information compression impossible, that way making such strings have a maximum size. From the same paper, Kolmogorov noted that Shannon entropy can be used, as one more approach of measurements of information complexity and mentions that there is no focus on the realization of Algorithmic Complexity and paper is more proof of concept. This is why later, such measure as expected Kolmogorov complexity was introduced (59). According to the research made by P. Grunwald and P. Vitányi. "Shannon information and Kolmogorov complexity" (59) expected Kolmogorov complexity equals Shannon entropy. This is quite close to their approaches, as Shannon's approach was focused on minimizing the number of bits, which are required to transmit a message.

Shannon entropy is calculated:

$$\mathrm{H(X)} = \sum_{x \epsilon \chi} p_x \log \frac{1}{p_x} \quad (1)$$

The same way is calculated Hartley and Natural information entropy. During calculation, it is possible to measure for single characters or combinations of consecutive characters (n-grams) as the amount of information is encoded into one bit.

2.3.2. Demographic bias

Demographic bias can be measured through the identification of the demographic parameters of the user/email. To achieve this, strings are split into n-grams and are fed to machine learning models, which based on the probability of the n-grams in word determines possible language, gender of the text. Some approaches simplify this task by taking only the last 3-4 letters of the word and based on them performing identification of the gender with such simple machine learning methods as Perceptron. However, as usernames and emails rarely have separators between words, such demographic identification problems are quite complicated. Additionally, to save resources and time, machine learning models trained by Google – Compact Language Detector v2 and Compact Language Detector v3 were used. Their architecture is explained in the Software section. Apart from the possibly detected language by model, this research used value from language detection models – prediction confidence. Prediction confidence can be correlated with the first characteristic of malicious users, as users try to stay anonymous, in that case, language detection models will have less confidence in language prediction.

2.3.3. Efficiency

While some malicious users try to be complex, other users try to be efficient. The efficiency of the malicious users during email and username generation can be measured by several calculations, which involve the fact of the user typing the text manually. This efficiency can be calculated by the unique number of letters in the word, as well as username length. Apart from those two, we can try to create features based on the typing behavior of the user, that includes different measurements.

One of the main characteristics of typing behavior is the distance traveled between each keystroke. To calculate this, we will create a mapping of the keyboards in a 2d array, with each key having assigned (x, y) coordinated, this way distance between keystrokes is the distance between 2 points. As keyboard keys are not perfectly aligned, different types of distances will be calculated. The most common distance calculation is Euclidean distance when the distance between 2 points is the length of line between them:

$$d(p, q) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2} \quad (2)$$

As it was mentioned, to have a better calculation of the distance, additionally we will use Manhattan distance, which is the sum of obsolete differences between two vectors:

$$d(p,q) = \sum_{i=1}^{n} |p_i - q_i| \quad (3)$$

Knowing those distances, it is possible to calculate total distance and statistical parameters, such as mean, median, and mode of keystroke distances in a given email/username.

Additionally, for typing profiling, it is possible to create finger mapping on the keyboard (see Fig. 6, where each color represents different fingers). Once we know the hand/finger used for each keystroke next metrics can be calculated: how often users prefer one or another hand/finger, how often consequently users used the same hand, how often users used the same or consecutive finger during the typing.

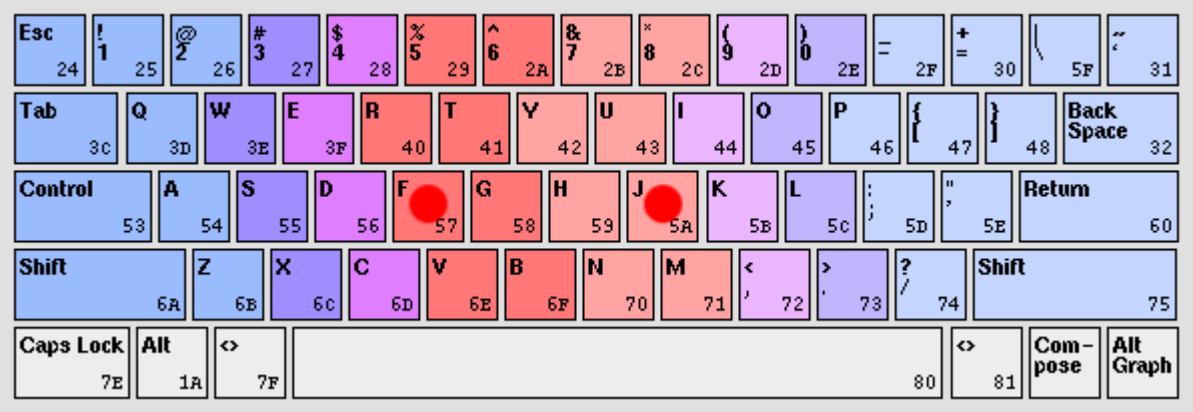

Fig. 6. Keyboard map for fingers (60)

Those metrics are not only limited by the hand/fingers, but it is also possible to segment the keyboard by zones or rows and apply the same calculations. To get the most out of those features, we can calculate the proportion of the same hand/finger used during typing.

As those calculations are based around behavior during typing, an optimal benchmark for comparison is needed. For that purpose, we used the most popular QWERTY keyboard layout (and one of the standard ones for modern PC) and compared it with the DVORAK keyboard, which was created based on a statistical analysis of the letter/word usage frequency (61). Such a comparison of efficiency between QWERTY/DVORAK could show malicious users, as they would look more efficient on the QWERTY keyboard, while good users would be more efficient on the DVORAK.

Additionally, efficiency can be measured by usage of convenience features in the email (i.e., case-sensitive typing, usage of tags/dots), because while being efficient malicious users rarely return to the old user identificatory, because they usually assume it was already compromised.

2.3.4. Similarity

There are different string similarity measurements, such as Hamming, Levenshtein, etc., however, to evaluate the similarity of the email or username, there is a constant need for database scanning, which is suboptimal because such need prevents scalability. Below is a brief explanation of the most popular distance metrics:

- *Hamming distance* compares two string and returns the amount of different corresponding characters in the string, can be calculated only for the strings of the same length.
- *Levenshtein distance* suggests how many edits of the string must be made, so the strings become identical.
- *Jaro distance* calculates the amount of matching character in strings when they are in the same and not farther distance than $\left\lfloor \frac{\max(|a|,|b|)}{2} \right\rfloor - 1$ where a and b are the characters. This solves the possible mistypes.

One thing, which is similar for those distance calculations, is that they are counting unmatched characters and calculate the number of operations or similarities to make strings identical. In this research we could try to calculate the similarity between good users (as our datasets are imbalanced towards the good user), that way being a good user, would mean that they are like each other. To measure that we could use concepts of string similarities, where we would try to find similar strings in population, by splitting all the usernames/emails into n-grams, calculating the probability of each n-gram appearing in our dataset.

Once we have probabilities of each n-gram, it is possible to calculate the probability of such username existence. We define, that event of each n-gram following the other one is independent, that is why we can calculate the total probability of such word using the multiplication rule of independent event $P(A \cap B) = P(A) \times P(B)$.
Once the probability of each email is calculated, it is possible to use information complexity theories to calculate the expected Algorithmic Information Criterion by Kolmogorov, which was reviewed earlier. For this, we would need to use Equation 1 with the logarithm of 2. In that case, usernames and emails which are likely to be more similar will have smaller information entropy.

2.4. Machine learning algorithms

During this research, we use supervised machine learning algorithms, which are designed for classification tasks. For the classification problem we define the following, that the model during training finds function *F(.)* which given the email or username as input "*x*", returns prediction if the user is malicious or not:

$$F(x) = \begin{cases} 1, & If\ x\ belongs\ to\ malicious\ user; \\ 0, & Otherwise \end{cases}$$

During this research, we will not prepare a featurization pipeline, that is why "x" will be given as input of calculated features for the given username or email address. Below we will investigate the algorithms which can be used for approximation target function F(x).

2.4.1. Logistic regression

A logistic regression model is typically formulated by relating the probability of some event *E*, occurring, conditionally on a vector *x*, of explanatory variables, to the vector x, through the functional form of a logistic cumulative distribution function (62):

$$P = \frac{1}{1 + e^{-(\alpha + \beta X)}} \tag{4}$$

Where P is the probability of 1, (α, β) are parameters of the model, estimated from the data. The value of α yields P when X is zero, and β adjusts how quickly the probability changes with X changing a

single unit (63). Because the relationship between P and X is non-linear – β does not give straightforward interpretation the same way it is in linear regression (63). This type of model is used to classify observation in one of the two populations, by letting *E* to give the probability of the event, that observation belongs to the first population and x being the attributes (in our case features), by which observation is classified into one of the populations (62).

In machine learning, logistic regression because of the large feature sets tends to overfit. To solve the problem of overfitting, there exist different types of regularization, which means that model gets penalized for features having big weights. The most common regularization methods are following:

$$L1\colon \mathrm{J}(\beta) = -\frac{1}{\mathrm{n}}\sum_{i=1}^{n}\left(\mathrm{y}_i \ln \mathrm{p}(\mathrm{x}_i) + (1-\mathrm{y}_i)\ln\left(1-\mathrm{p}(\mathrm{x}_i)\right)\right) + \frac{\lambda}{2\mathrm{n}}\sum_{j=1}^{d}\left|\beta_j\right| \tag{5}$$

$$L2\colon \mathrm{J}(\beta) = -\frac{1}{\mathrm{n}}\sum_{i=1}^{n}\left(\mathrm{y}_i \ln \mathrm{p}(\mathrm{x}_i) + (1-\mathrm{y}_i)\ln\left(1-\mathrm{p}(\mathrm{x}_i)\right)\right) + \frac{\lambda}{2\mathrm{n}}\sum_{j=1}^{d}\beta_j^2 \tag{6}$$

Although the formulas look similar, those regularizations have a few key differences. During the training process *L1* penalty shrinks the less important features to zero, that way removing them during feature selection, while the *L2* removes a small percentage of the weight from features on each iteration and never converges to 0, that way – all features are in the approximated model function.

2.4.2. Support Vector Machine

This algorithm was first introduced in 1992 by Boser, Gyon, and Vapnik in COLT-92. This model algorithm is called a generalized classifier, which means that like all other classifiers, its target is to maximize prediction accuracy while avoiding overfitting. The requirement for this method is, that there exist two different classes of observations and they can be separated by one hyperplane. This means, that there can be an unlimited number of hyperplanes, which could separate those classes. Assuming that there are no outliers, the Support Vector Machine (SVM) algorithm tries to find the hyperplane, where the distance between the hyperplane and closest observations of each class is maximal (Figure 7). That is the reason, why SVM is called "Maximum Margin Classifier" by some of the sources.

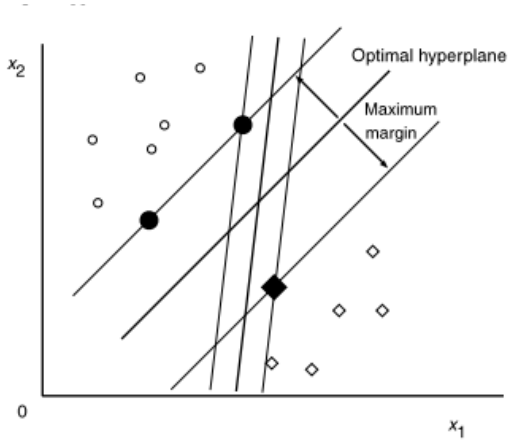

Fig. 7. Optimal separating hyperplane in two-dimensional space (64)

In general form, function approximated by SVM algorithm, when $\alpha > 0$:

$$D(x) = w^T x + b = \sum_{i \in S} \alpha_i y_i x_i^T x + b \quad (7)$$

There exist different modifications of the SVM algorithm, based on the type of margin: hard margin, soft margin – i.e., when we define, that some observation can be closer to the hyperplane or overlap. Additionally, it was noticed, that when the observations are not linearly separated between classes – it is possible to exchange linear function with symmetric function, which was named as "kernel" function. That way changing the kernel, it is possible to distinguish classes not only for the observations, which are separated by a linear function (Figure 8).

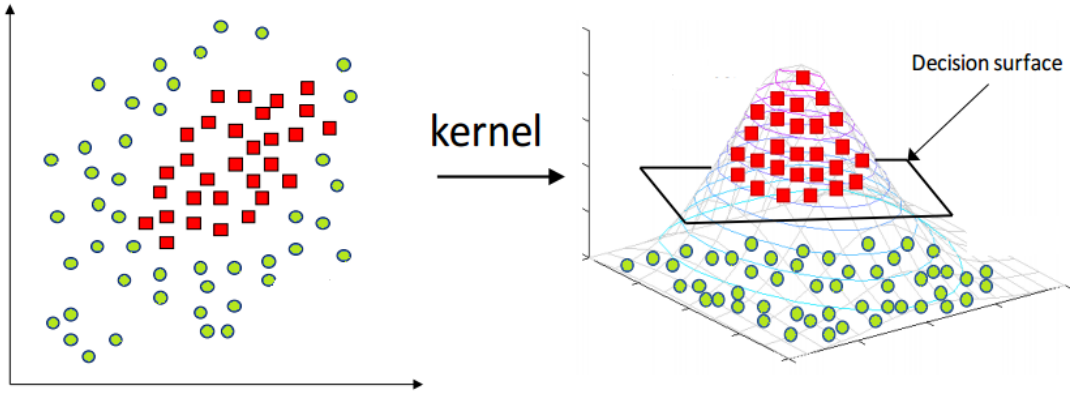

Fig. 8. SVM kernel trick (65)

2.4.3. Decision tree

Decision tree algorithms are representatives of the supervised classification algorithms, which can be tailored for regression problems as well. The purpose of the classification algorithm with decision trees is to create a model, that predicts the target variable. A decision tree can be visualized graphically (Figure 9) as a flowchart structure, where internal nodes represent features or attributes, branches are decision rules (comparison operations) and terminal node or leaf represent the outcome. Such visualization mimics the decision-making process of the human, and that is why can be interpreted, and we can interpret those features, which are on the top of the tree are the most important ones.

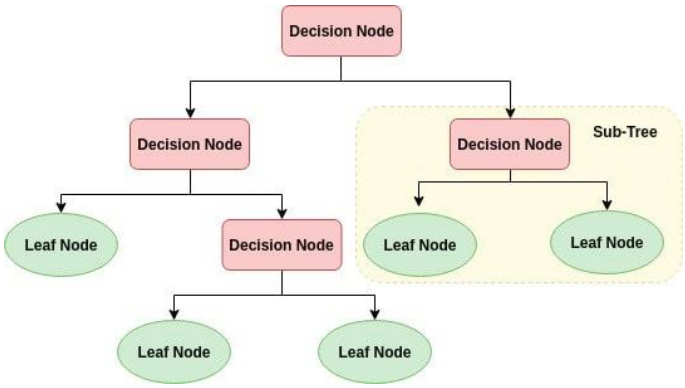

Fig. 9. A decision tree with terminology (66)

Decision tree building algorithm consists of next steps:

1. Selection of the best feature for splitting based on Attribute Selection Measure (ASM).
2. Generation of the branch, to split the dataset into smaller subsets.
3. Return to the first step for each child, until stopping conditions are met (i.e., there are no more observations to classify, or all variables used and there is no possibility to create new branches).

There exist different versions of the algorithm realization, based on different calculations of ASM, so those versions differently find nodes and split branches. For example, the ID3 algorithm uses information entropy and selects nodes and branches based on the maximization of information gain after the split. The downside of this algorithm was, that based on information gain, the algorithm was prioritizing features, which had the most unique values. To avoid such bias, new versions of the tree-building algorithms were introduced, such as the calculation of normalized information gain or class impurity.

The biggest disadvantage, however, is that decision trees tend to extreme overfitting, and that is why the different methodologies how to prevent this were introduced, such as pre pruning and post pruning, but is very hard to find the optimal point for it. However, those problems are solved when decision trees are united in model ensembles, which will be reviewed below.

2.4.4. Random Forest

Random Forest is the algorithm, which as denotes in its name is the ensemble of the decision trees. During the declaration of the algorithm, the final number of the trees is set up, and the algorithm tries to create an ensemble of the randomized trees. To achieve the creation of independent trees, the algorithm uses the next methods:

1. Bagging (bootstrap aggregation) – as decision tree building highly depends on the data, if data is sampled with a replacement for each tree, this will result in the creation of different trees.
2. Feature randomness - during the split of the decision tree, every feature is weighted during a split, however, if for each tree there is a limited different set of features, trees can not use the same features for splits. This creates more variation between trees in the model and helps to create uncorrelated decision trees.

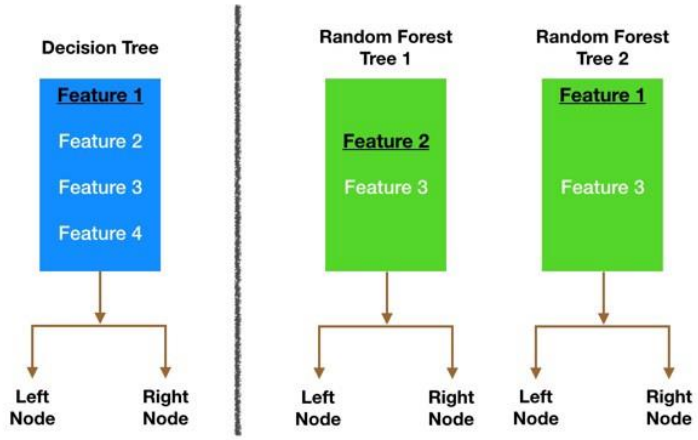

Fig. 10. Features for splitting in the Decision tree and Random Forest (67)

After the forest is created, model prediction is based on the majority votes. This gives an improvement over the regular decision tree algorithm if the trained trees are uncorrelated between themselves. The idea behind this, that error of prediction of each tree in the ensemble is corrected by other trees. Although Random Forest is more resistant to overfitting comparing to the Decision Tree algorithm, and it is easier to control it, as the number of trees and complexity of trees can be set up during training algorithm initialization.

2.4.5. Gradient Boosting

Gradient Boosting (GB) is the type of algorithm when weak learners (which can predict slightly better than random chance) are modified to become better. The goal behind was described by Michael Kerns in "Thoughts on Hypothesis Boosting" (68) as "an efficient algorithm for converting relatively poor hypotheses into very good hypotheses". The early implementation of boosting was the AdaBoost algorithm, which uses decision trees with a single split. AdaBoost uses weights, to add more weight for complicated classifications of observation and giving less for those, which are easy to detect. Trees are created in order by complexity, from the simple ones, sequentially adding new ones to detect more complicated patterns. Predictions are made by majority voting, accounting for each learner's weight. Later other algorithms were introduced, which are using more complicated algorithms with a loss function.

In the general case, gradient boosting has three elements:

1. Loss function. Usually decided by the problem, however, it must be differentiable. As in this research project binary classification is used, we are using binary log loss function:

$$Log\ loss = \frac{1}{N}\sum_{i=1}^{N} -(y_i \times \log(p_i) + (1 + y_i) \times \log(1 - p_i)) \quad (8)$$

2. Week learner. Although AdaBoost uses single split decision trees, other algorithms may utilize more complicated trees, which can increase the accuracy of predictions.
3. Additive model. During the model training, one tree at a time is created and older trees are not changed, through the utilization of gradient descent loss is minimized when new trees are added.

In 2019 Kaggle.com surveyed data scientists to gather data about the most popular algorithms, 37% of participants responded, that they use Gradient Boosting algorithms on daily basis, such as XGBoost and LightGBM (69).

2.5. Model performance evaluation

After machine learning is complete, there is a need to evaluate the performance of algorithms. First, for this purpose, we randomly split the dataset into train/test datasets (80:20). Such an operation allows us to test the model on an unexplored dataset.

For accuracy and error measurement, as we are using binary classification (there are only two possible classes) we have four different outcomes: the user was good, model evaluated it as good (True Positive); the user was good, model evaluated it as bad (False Negative); the user was bad, model evaluated it as good (False Positive); the user was bad, model predicted as bad (True Negative). This is better visualized in Table 1.

Table 1. Confusion matrix used for result validation.

		Predicted Condition	
		Predicted Condition Positive (PP)	**Predicted Condition Negative (PN)**
Actual condition	**Actual condition positive (P)**	True Positive (TP), hit	False Negative (FN), Type II error
	Actual condition negative (N)	False Positive (FP), Type I error	True Negative (TN), correct rejection

After we define different labels for different outcomes of prediction, we can calculate the metrics of each model and compare them. Following metrics and techniques can be used for model comparison:

- Receiver Operating Characteristic (ROC) graph – technique, which visualizes classifier performance, based on True Positive Rate and False Positive Rate plotting on different probability thresholds. It has numerous advantages, one of which is disregard of class distribution or error costs (70). That is why it is convenient to use ROC with an unbalanced dataset.
- Area-under-Curve (AUC), which is based on the Receiver Operating Characteristic (ROC), its interpretation is that it is the probability, that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (70).
- Sensitivity (True Positive Rate/Recall) – measures the rate of correct prediction of the positive class, in other words – how many positive observations classified as positive from the total positive observations:

$$Sensitivity = \frac{TP}{P} \tag{9}$$

- Specificity (False Positive Rate) – measures the rate of the correct prediction of the negative class or how many negative class observations were correctly classified of total negative class:

$$Specificity = \frac{TN}{N} \tag{10}$$

Based on those proposed metrics and graphs there is a possibility to select the most favorable models and review their structure.

Additionally, trained models can be revalidated to check if their outputs are independent. Statistical comparison was selected, which is present in a tool named easyROC. It uses Bonferroni multiple comparison test with DeLong's test, where p-value < 0.05 is considered as a statistically significant difference.

2.6. Used software and libraries

The software used in this project is Jupyter Notebook with Python 3.8.5. Jupyter Notebook provides an iPython environment, which gives a convenient workflow for step-by-step analysis. Python was favored over R, SAS, as there is currently a wide variety of possible libraries for data analysis and machine learning model creation in Python, their codebase is unified and there are almost no problems

in dependencies between different libraries. Most of the libraries used in this research were added based on the project pipeline.

During data preparation and feature generation the following libraries and its version were used:

- NumPy 1.19.2 – a library provided by an open-source project Numpy.org. It is a framework for n-dimensional arrays (matrices) and operations/calculations with them. Its core functionality is implemented in C/Fortran, which means, that calculations are performed swiftly compared to pure Python and most of the functions are well documented and have a clear and elegant declaration.
- pandas 1.1.3 – open-sourced library, which is developed by more than 2000 contributors around the world. It provides the framework of DataFrame, which implementation is fast and rich with functionality. The core of the library is written in C/Cython, that is why the speed of data manipulation outperforms pure Python (same as numpy). It allows reading and writing data from numerous file formats, automatic data alignment, and integrated handling of missing data. Data slicing and other operations are optimized for performance, that is why it is very convenient to use the DataFrame framework from this library for large datasets.
- re 2.2.1 – Regular expression operations library. This library provides operations with strings and bytes-like functionality in the Perl programming language (71). As its implementation does not require full string capture for performing pattern search and replacements, computations with this library are efficient and suitable for big amounts of data.
- nltk 3.5 – a library that poses itself as a toolkit for natural language processing. The package has a lot of different methodologies and approaches for language processing, however, the main purpose during this project was splitting strings in n-grams, for feature calculation, as its implementation is faster compared to pure Python.
- pycld2 0.41 and cld3 0.22 – both are Python implementations of Compact Language Detector by Google. Those libraries give estimates of which language string is written. Predictions are made by the neural network model which architecture can be found in Figure 11.

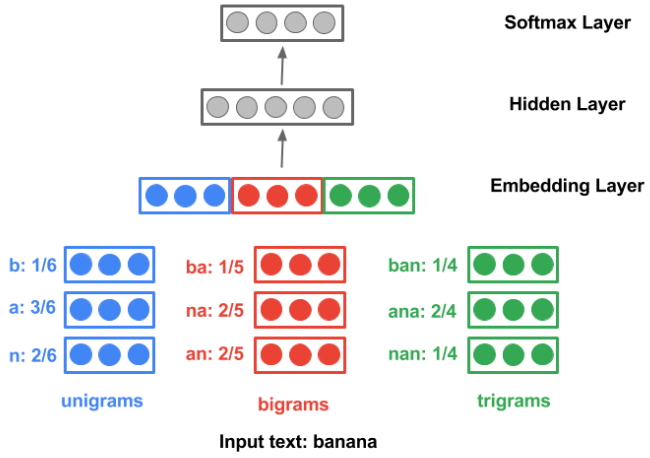

Fig. 11. CLD v3 model architecture (72)

- mapply 0.1.7 – library dedicated for parallel computing in Python. During feature engineering, it is required to fully scan DataFrame and this library adds multi-threading support for in-built Python methods and that way makes calculations more efficient.
- Seaborn – visualization library for Python, it will be used during Exploratory Data Analysis of the generated features.
- Scikit-Learn – Machine-Learning framework, which is built on NumPy, SciPy, and matplotlib libraries. It has different tools for machine learning, which help to preprocess datasets, train models, and interpret the results of training.
- XGBoost – optimized for Python gradient boosting library focused on XGBoost algorithm, allows multi-threading that way making training routine of the Gradient-Based models effective. It can use Scikit-Learn API, combining the robustness of its implementation and the possibilities of the Scikit-Learn library.
- LightGBM – one more optimized for Python library, tailored for LightGBM algorithm, which can also use parallel computing during the training process and utilizes Scikit-Learn API. It is possible to create own loss functions and use them during training and evaluation of models. Additionally, the library is more optimized for large datasets, that is why training time comparing to XGBoost can be up to 7 times faster.
- easyROC – an interactive web tool for ROC curve analysis (73). This tool consists of different methodologies for Receiver Operator Characteristic curve analysis, of which in this project will be used multiple comparisons between classifiers, to identify the independence of predictions.

2.7. Second section summary

During this section, we overviewed the problem domain and how training/testing data will be gathered. Additionally, we investigated the methodology for feature engineering and overviewed calculation concepts. As we will train several models, used in the evaluation, their metrics were explained. We overviewed the most popular supervised classification algorithms and selected the ones which are used during the research. Lastly, we prepared a list of the software and Python libraries, which are used by the software during the project.

3. Research results

This part of the thesis is aimed to apply reviewed methodology during feature engineering. After features are calculated, different models are trained and results are analyzed.

3.1. Project pipeline

According to the project goals and raised problems, the proposed structure of the research is following:

1. Dataset reading and labeling.
2. Feature engineering
3. Exploratory data analysis of calculated features
4. Supervised model training
5. Model result interpretation
6. Discussion

3.2. Software and hardware

As it was mentioned during methodology, the used programming language was Python with Jupyter Notebook iPython environment with libraries specified in the Second Section of the project. Used The used operating system is Ubuntu 20.04, which is operating under Windows Subsystem for Linux (WSL2). Used hardware – CPU Intel i5-8300H, 16GB RAM, SSD hard drive. Such hardware specification allows swift data reading, feature generation, and model training.

3.3. Data labeling

As it was mentioned in the methodology section – email addresses and usernames for the dataset were collected from the data breaches and datasets containing Twitter communication. However, in order to prepare the dataset, the following requirements were raised for the usernames and the email addresses.

3.3.1. Email dataset labeling criteria

For the good email addresses:

- Data breach must be from the service, where users prefer to use main emails, not throwaway ones. As an example – car rental service, governmental service, etc.
- If the data breach is from the e-commerce website – products should not be highly liquid for fraud and the size of the dataset must be at least 10 000 email addresses. Definition of liquid products is that they have high demand in the market and can be easily resold without traces (popular electronics, digital content)
- Region of the data breach – during dataset creation of emails, it is better to use datasets of services that operate in the region where detection is targeted, because depending on language different linguistic rules apply. As this is the initial research, that the best idea is to collect all datasets for the good users from the same regions.

For the bad email addresses:

– The email was reported at public or private dataset as fraudulent, should not be duplicated in the good email dataset.

3.3.2. Username dataset labeling criteria

For the good usernames:

– The user must have a positive number of followers (more than 50).
– Is not part of any spam dataset.
– Verified by Twitter or matches the criteria above.

For bad usernames:

– Malicious links were noticed in tweets.
– Is part of the botnet, which sends tweets.
– Suspended by Twitter usernames based on the malicious activity.

3.4. Collected datasets for research

According to the methodology and criteria defined in the Data labeling subsection – the following datasets were selected for the research:

– CityBee 2021 data breach, belongs to the Lithuanian car rental service. Contain 110302 rows; columns include – email address, hashed password, name, surname and personal identification number. All data columns apart from emails are removed.
– The leak of German emails, supposedly belonging to the beauty care retail store data breach, contains a so-called combo of email and passwords, in total 61128 rows. The password column is removed and only email addresses are used.
– Leak called mail access, source unknown, however as the author of leak states, they belong to the real people and geographically mails are from France, Germany, United Kingdom, and other European countries. In total 34656 rows. Contains 3 columns – email, unencrypted passwords, and IMAP server addresses. Based on the last parameter, possibly one of the email clients was compromised. As research is focused on the buildup – all irrelevant to the research information was removed.
– Enron spam email dataset – this dataset was provided by Enron for data scientists to train Natural Language Processing models. Every document in this dataset consists of an email message. Provided messages are classified as spam or ham, where spam is considered either unwanted advertisement email or attempts of social engineering. In this research only spam class emails are used, for email address information only reply-to email is used (as this is guaranteed, that this email is in possession of a malicious user). In total 2095 email addresses are collected. If all mentioned in dataset email addresses we used – there would be twice more, however, in order to maintain a high-quality dataset, this was not used.
– Cleantalk.org parsed dataset from “recently updated lists” – 1971 emails. Those emails were reported by websites using Cleantalk.org spam prevention plugins as malicious and published on the front page of the Cleantak.org website.
– Cleantalk.org paid dataset, where emails are published based on the frequency of spam reports. The selected paid version has emails, which were reported over 200 times. In total there are 47667 email addresses, with additional information about the record – how many

times the email address was reported during the last week, how many times in total, the first time appeared in the database, and the last time entry was updated.
- Spam email dataset published on Reddit – 3038 emails only dataset. According to the description, this dataset consists of emails, which were noticed during the attempts of affiliate program reward abuse (however, there is no guarantee, that all email addresses were generated by malicious users).
- Twitter dataset archive is represented as collections datasets consisting of API responses from Twitter (Data structure is represented in Appendix 1). From these archives, good and bad users are separated based on the topics and user info is used in further research.
- The Twitter dataset provided by https://voterfraud2020.io where it is possible to filter out users, which were suspended/deleted because of their "Vote Fraud" activity. Additionally, this dataset has a good user list, which could be counted in the good user dataset for future researches.

In total, for this research we managed to acquire the following amount of the labeled usernames and emails after removing duplicates, as some of the emails/usernames were repeated in several datasets:

Table 2. User counts by entry type and classes

Entry type	Good user	Malicious user	Total
Email	201008	54345	255352
Username	10523	3051	13574

As it can be observed – datasets are slightly imbalanced (there are more good users, comparing to the bad users), that is why the following techniques may be used during model retraining – usage of class weight, limiting maximum delta step, which regularizes, how boosting and training is performed, that way making possible to find better models.

3.5. Feature engineering and exploratory data analysis

After datasets are prepared for feature engineering, the first thing, which is examined – the popularity of email address local parts and used domains by good and bad users. As it can be seen from Table 3, both good and bad users have their favored email address providers. Possible reasons – if the good user registers with a free email provider – he prefers email providers, which have rich functionality and long history. In general, those providers try to maintain a high profile and fight automated registrations as this harms good username as well (less available local parts). At the same time, malicious users favor email providers, which have faster registration, and sign-up can be automated (there are no complicated captchas). In collected datasets, there was not observed massive usage of temporary email providers.

Table 3. Most popular email providers and their popularity among different types of users.

Domain name	Total users	Popularity, total	Good users	Popularity, good	Malicious users	Popularity, malicious
gmail.com	89589	35.08%	84625	42.10%	4964	9.13%
t-online.de	36427	14.27%	36424	18.12%	0	0%

mail.ru	15674	6.14%	1042	0.5%	14632	26.92%
bk.ru	8498	3.33%	67	< 0.01%	8431	15.51%
interia.pl	7702	3.02%	7684	3.8%	0	0%

Popularity is the metric, which represents user preference for certain email providers (number of users using certain providers divided by the total number of users). As mentioned earlier some of the email providers are heavily targeted by malicious users. It is always possible to limit the usage of certain providers for the service (i.e. do not allow registration from mail.ru domain), however for malicious users switch of provider is performed easier comparing to the good users can select others, while for a good user this may become an issue.

It is noticeable, that most of the bk.ru and mail.ru come from the biggest data source for the malicious user – CleanTalk.org paid dataset.

Table 4. CleanTalk.org paid dataset.

Email provider	**User count**
mail.ru	12661
bk.ru	8411
inbox.ru	7468
list.ru	7027
gmail.com	4008

Apart from “mail.ru” and “bk.ru”, there is also a large number of emails having provider names – “inbox.ru” and “list.ru”. All those email domains are assigned to one service provider – Mail.ru Group, which is why, supposedly, this email provider has the easiest registration flow. Additionally, one of the theories, that malicious users are demographically biased is being confirmed.

If we would check different local parts of the email, how often they reoccur, we may notice, that bad users do not share the same local parts, while good users often have the same local parts.

Table 5. How often good and bad users use the same local parts of the email.

Good user local parts	**User count**
info	1143
tomas	208
mindaugas	148
andrius	142
darius	116
mantas	106
marius	103
service	100
paulius	83
tadas	77

Bad user local parts	**User count**
info	71
kevin	19
robert	17
admin	15
noreply	13
steve	10
support	10
sales	8
qwerty	8
eco	8

As it is possible to observe, good users tend to use their name, although we tried to use not only the Lithuanian dataset, as it is also seen, that Lithuanian users prefer to use their first name as the local part. Other than that, there the leader of the recurrence in both user types – quite standard for non-free domains "info@" email local part.

3.5.1. Typing behavior-based features

Such a name is granted, as those are the features that are based around the typing behavior of the users when they generate their username or email. Most of the concepts are mentioned in the methodology section, defining how it is possible to measure the efficiency of the user.

For calculation, as it was mentioned, we will use two sets of keyboards – QWERTY and DVORAK. We created the map of each keyboard as two-dimensional arrays. After the creation of those arrays, it is possible to create typing heatmaps, however, as it is hard to find a proper bitmap of DVORAK for heatmap creation in Python, the following website was used - http://patorjk.com/keyboard-layout-analyzer/#/main. As we can see typing behavior for good and bad users differ, especially it is noticeable when we use the DVORAK keyboard for the benchmark.

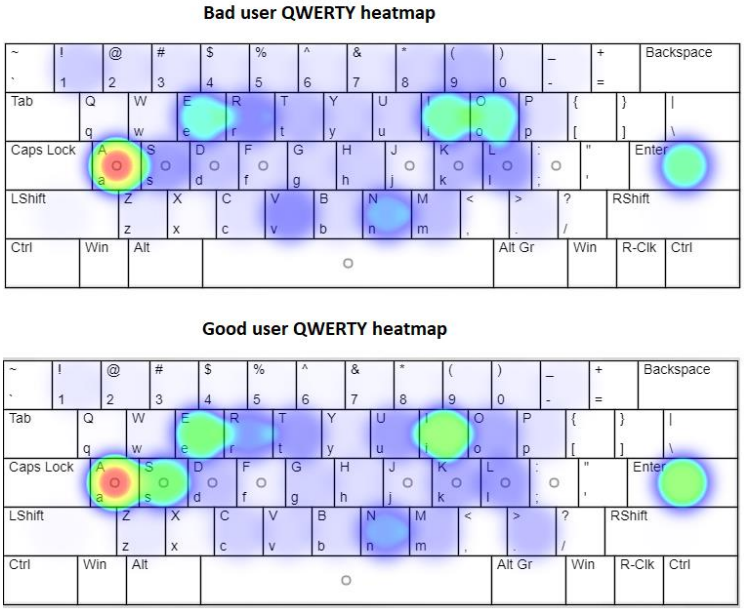

Fig. 12. Email typing heatmap with QWERTY layout. Malicious users vs regular.

From the QWERTY heatmaps of the email local parts (Figure 12), there are certain trends observed, bad users tend to use more numbers, comparing to the good users, although top letters seem to be similar, there are still key differences. One more observation is that good users tend to use a less letter F but use more dots. If we look at the bottom row of the keyboard, it is quite noticeable, that bad users are avoiding using letters from it, and when they use, they use primarily letters that are accessible with the forefinger. As we mentioned, for benchmark, we are using DVORAK keyboard layout.

As it is seen from DVORAK heatmaps (Figure 13), we can notice some trends as well. As DVORAK was designed around probabilities of the used words and letters in them, it aimed to create an ergonomically comfortable keyboard, utilizing equally both hands, with a primary focus on the

middle and top row. For good users, we can see this trend maintaining, while for the bad users – right-hand ring-finger and pinky must type more frequently on the bottom row more comparing to the good user layout. In the top 10 keys frequencies, there is a slight difference but are not as significant as it can be seen from heatmap visualization, except for some letters such as “T”, “S”, “O”, “U”.

Table 6. Frequency ranked of typed letters by bad and good users.

Key	Good user rank	Bad user rank	Key	Good user rank	Bad user rank
A a	1	1	R r	6	7
I i	2	2	T t	7	**11**
S s	3	**9**	U u	8	**18**
E e	4	5	L l	9	9
N n	5	6	O o	10	**4**

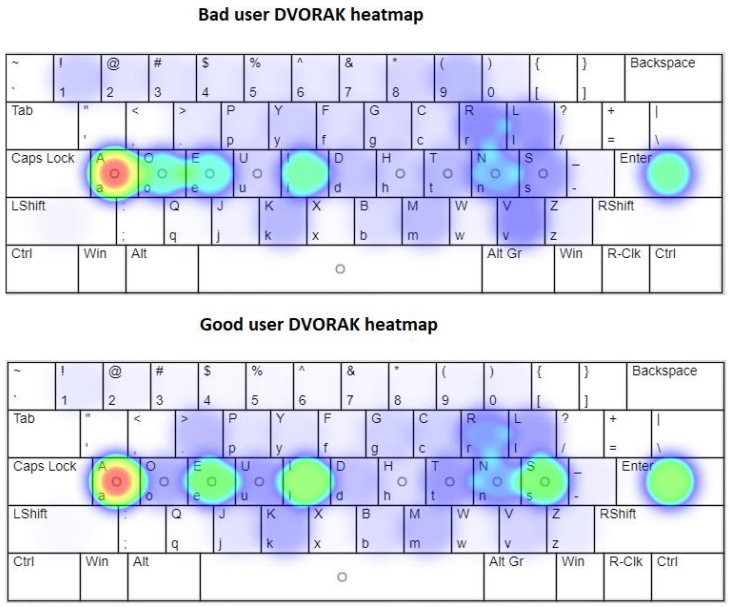

Fig. 13. Email typing heatmap with DVORAK layout. Malicious users vs good users

Comparing the Twitter username heatmaps (Figure 14), there is little to no observable difference between heatmaps as it is seen with email datasets. There are several differences in frequency ranks, but they are not as significant compared to the email datasets. A possible explanation is that with email local parts users do not have such harsh requirements, comparing to Twitter, where there is a symbol limit, and each username has to be unique. However, by the intensity of the numeric row, it is seen, that bad users use a little bit more digits.

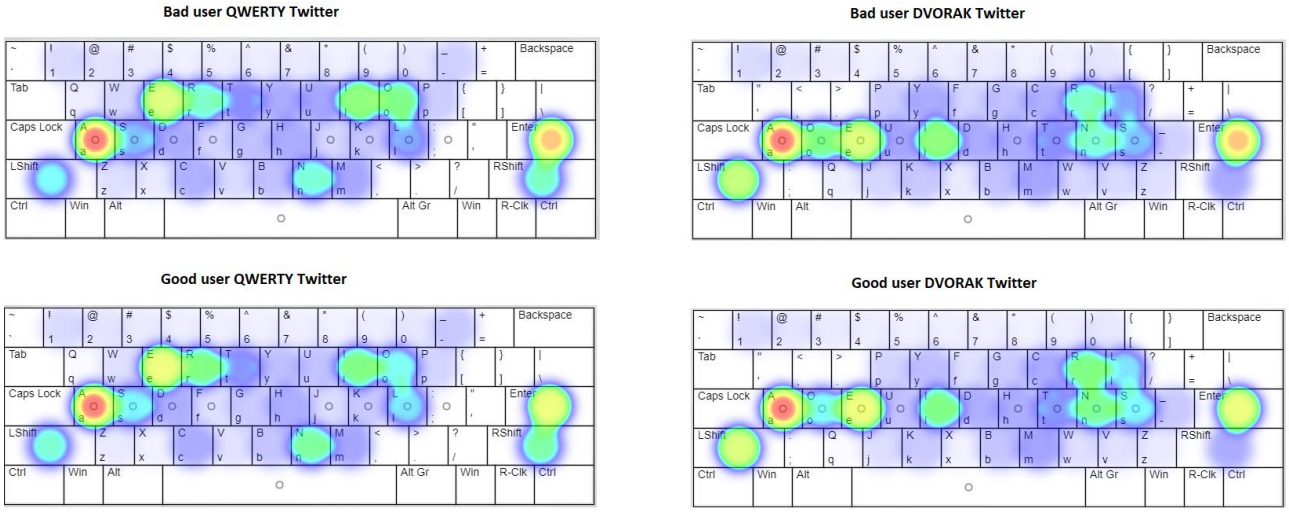

Fig. 14. Twitter dataset typing heatmaps. Malicious users (top row) vs good users (bottom row)

Based on the observed differences and additionally to measure how hands and fingers are moving during typing process, the following feature sets were created labeled as “feat_typing”:

- Distance metrics between consecutive keystrokes. To achieve this, each local part or username is split into bigrams, and distances are calculated and put in the array (Euclidean, Manhattan, and Log-Euclidean to normalize scales). As we want to deal with structured data, distances from arrays are aggregated with those functions – sum (total distance traveled), mean (average distance between keystrokes), median (50% of keystrokes do not exceed that distance), and mode (most repeating distance). In total, as we are evaluating those distances for 2 keyboard layouts, we have 24 features.
- Hand and finger usage, as it was noticed on heatmaps, we map each keyboard keys by hand and fingers and calculate the following metrics: strokes made by each hand, percentage of each hand used during typing, strokes made by each hand’s finger, percentage of strokes made by each finger, consecutive strokes made by the same finger or following one, percentage of the key strokes made by the same finger in the username or local part of the email. After calculations, we received 36 different features involving the mapping of hands and fingers and user behavior during typing.
- Keyboard row usage. For both layouts, there were calculated for each row, how many times it was used during formulation of email/username, percentage of characters in each row. Additionally, the same way how it was calculated for hands and fingers – calculations were made for consecutive usage of the same or neighboring row. In total – 24 features were created.

Those features are calculated both for Email and Twitter datasets. As we can see from distribution plots for the email dataset (Appendix 2), most of the distributions appear like a normal distribution from their graphs, however, without additional normality text we cannot claim so. From the email dataset typing feature boxplots by variable (Appendix 2), it is seen, that some variables have different distributions between good and bad users. One interesting example is the mode of Euclidean distance between keystrokes. It can be observed, that for the QWERTY keyboard, malicious users press keys, which are not far from each other (comparing with good users), with 75% of the most common distances between keystrokes being less than 4 distances (while good users have around 4.5) and 50%

of most common distances between keys not more than around 2.2 (comparing to 3 distances for good users), while on DVORAK keyboard, 75% of most common distances between keystrokes are less than 7 distances (comparing to 6 for good users) and 50% of both good and bad users have.

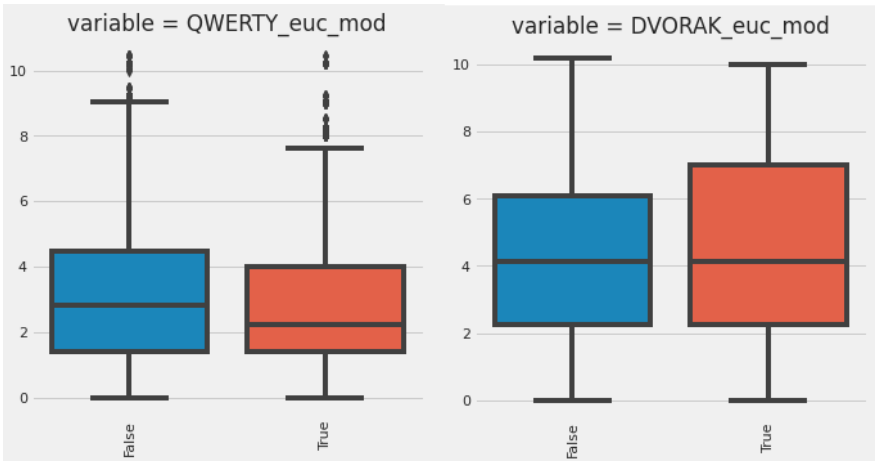

Fig. 15. Euclidean distance mode for email distribution between two types of keyboards.

In the case of Twitter usernames, we have similar distribution between features (Appendix 2). As most of the variables are around normal distribution, during training, this feature set will not need to get normalized for methods, which are sensitive to skewed data. After visualizing boxplots by variable (Appendix 2), it is seen, that observation distribution does not differ that much with features as they do with the email dataset. The only slight difference, which is noticeable – left-hand usage (Figure 16).

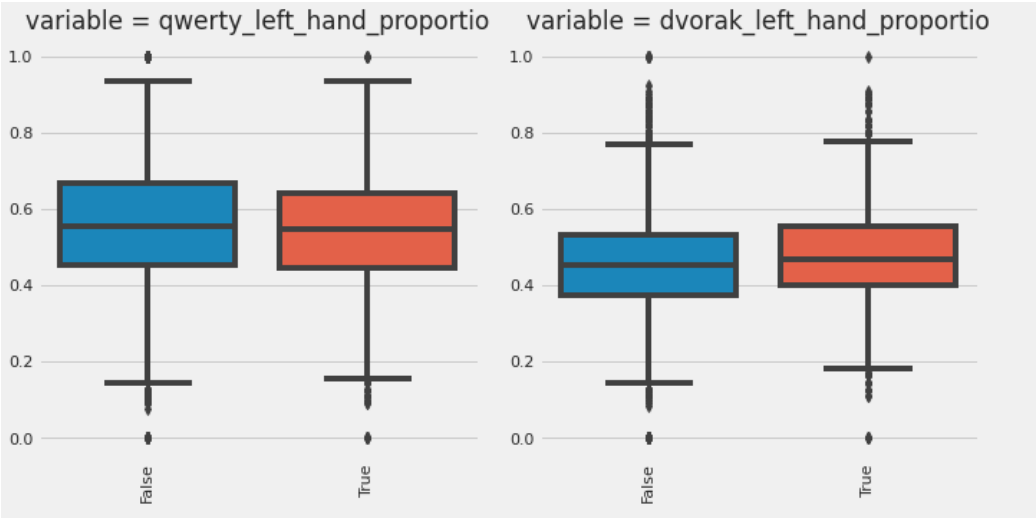

Fig. 16. Boxplots by the value of left-hand usage proportion feature for Twitter usernames

3.5.2. Information-based features

For information-based features, we performed several calculations of information entropy for each username and email part. As mentioned during the Methodology review, we will use Natural/Hartley and Shannon entropies measures. Shannon entropy we will use not only for text strings but for splitting text strings into n-grams. One more metric calculates is normalized Shannon entropy for n-

grams, where we normalize it by dividing from the information entropy of the same length string having all characters different (maximum information chaos). In total 7 features were calculated.

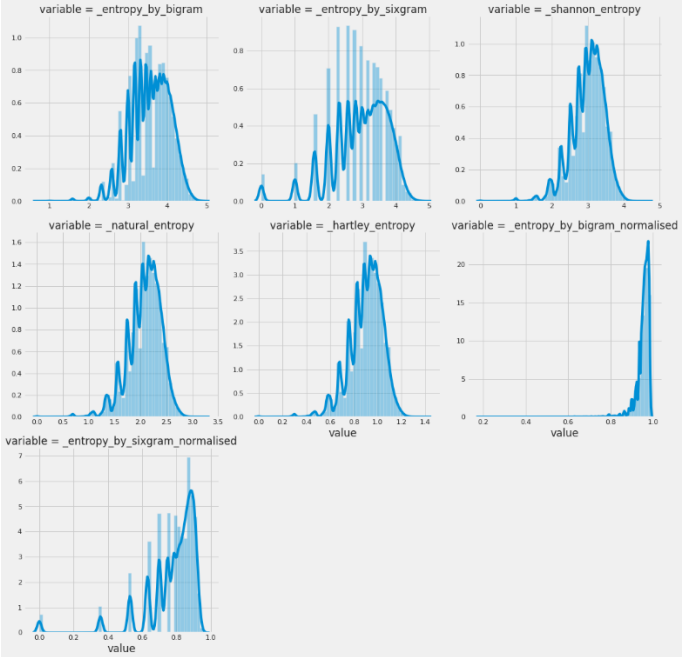

Fig. 17. Distribution plots of information feature for emails.

From distribution plots (Figure 17) of information features for the email dataset, we can observe that most of the information distribution plots look-alike normal distribution with one exception – normalized. For plotted boxplots by variable (Figure 18), we can notice that emails belonging to malicious users have a slightly higher informational entropy amount, which confirms the theory from the Methodology section, that malicious users try to be diverse.

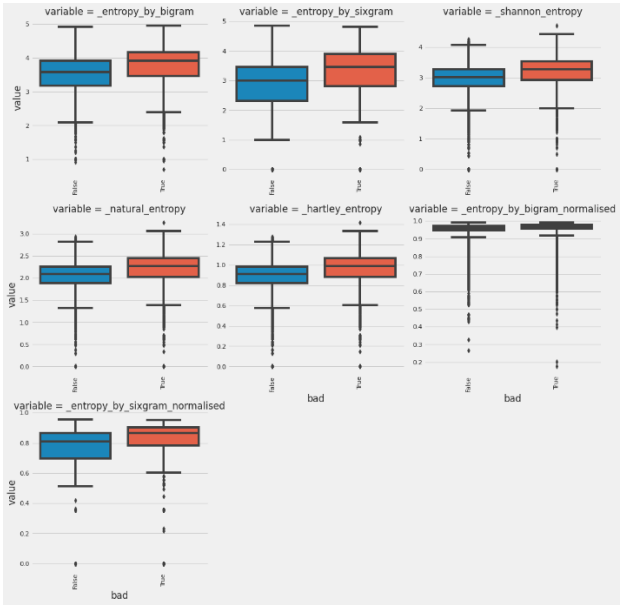

Fig. 18. Boxplots by variables of information feature for emails

Running the same featurization for the Twitter username dataset and later performing visualization of distribution plots and boxplots by variable (Figure 19) we can observe that in the Twitter dataset username information features have a little skewed distribution and distributions both for fraudulent and non-fraudulent features is quite similar, however, without statistical tests, it is not certain.

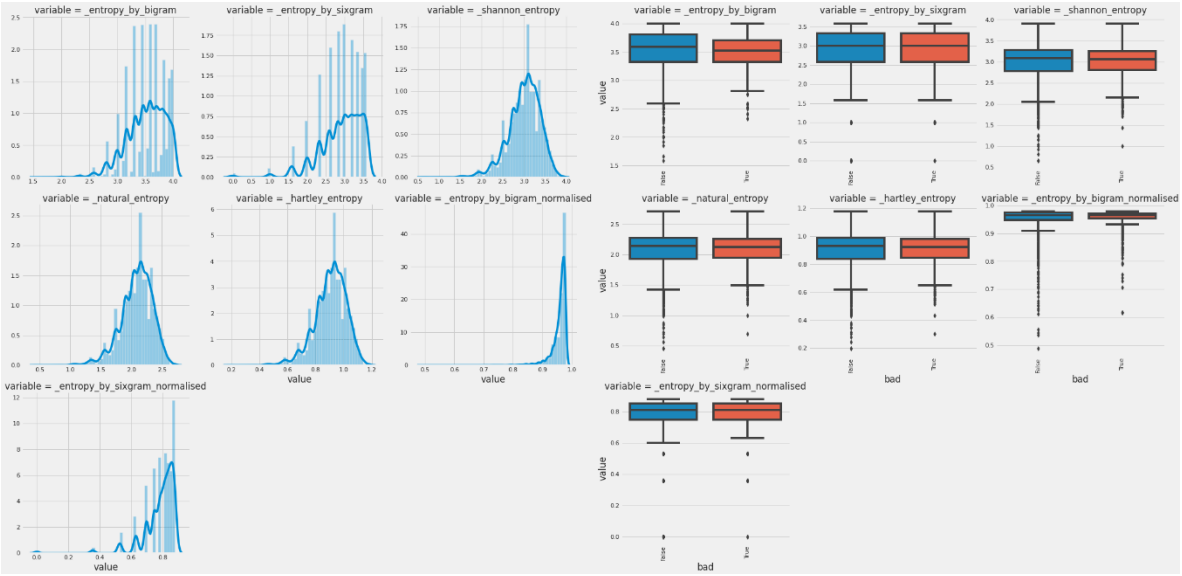

Fig. 19. Information features distribution and boxplot by variable plots for Twitter data set.

3.5.3. Similarity with information entropy

For this set of features, we calculated for each n-gram, which is met in the dataset probability of appearing in the word. Later based on those probabilities, we are calculating the probability of the word to appear and apply Shannon entropy to this calculation. This way similar entries will be matched with similar entries and we will be able to find the most similar entries because the chance of the word being made of those n-grams, will give the word the smallest possible entropy. In total 6 features were calculated, half of them are first measures divided by the given length of the string in the attempt to normalize results.

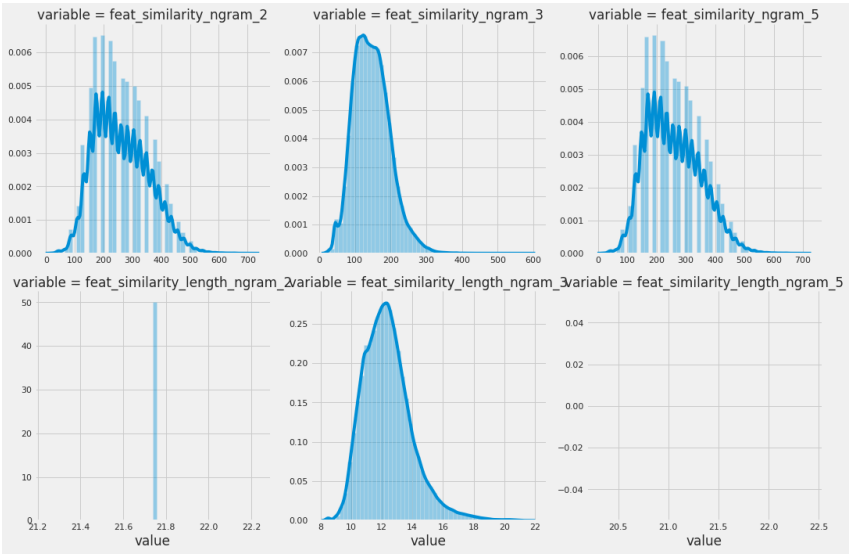

Fig. 20. Distribution plot for entropy-based similarity features for email dataset.

For the email dataset, we can observe from distribution plots (Figure 20) that distribution is quite skewed, and from boxplots (Figure 21) it is noticeable, that malicious users have a little higher value of such information entropy, comparing to the good and this shows, that good users use more common similar names/words, comparing to a malicious one.

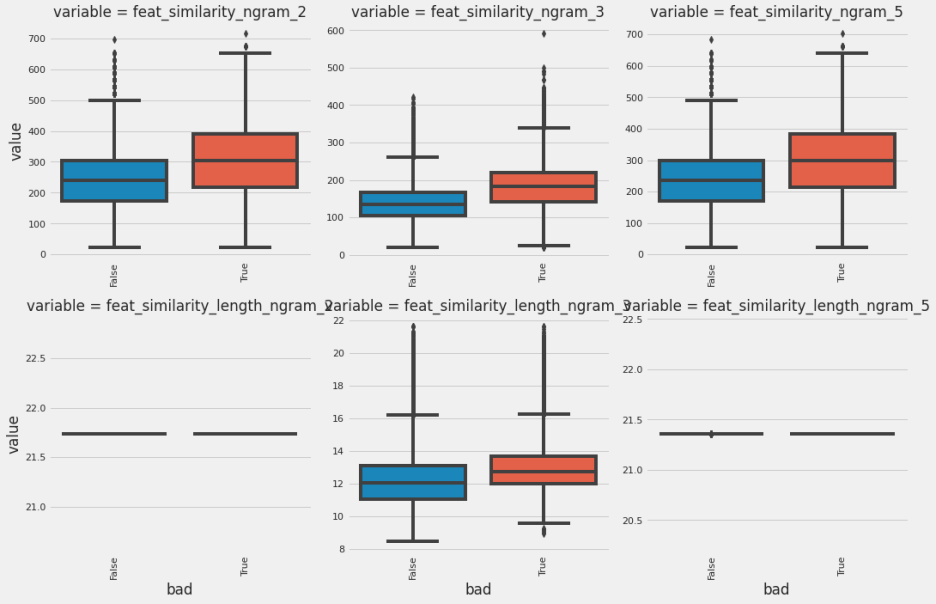

Fig. 21. Boxplots by value for entropy-based similarity features for email dataset

For the Twitter username database, however, we observe (Figure 22) the same trend as with previous feature sets, that the calculated feature distribution is very similar between bad and good classes. This might make detection tasks harder.

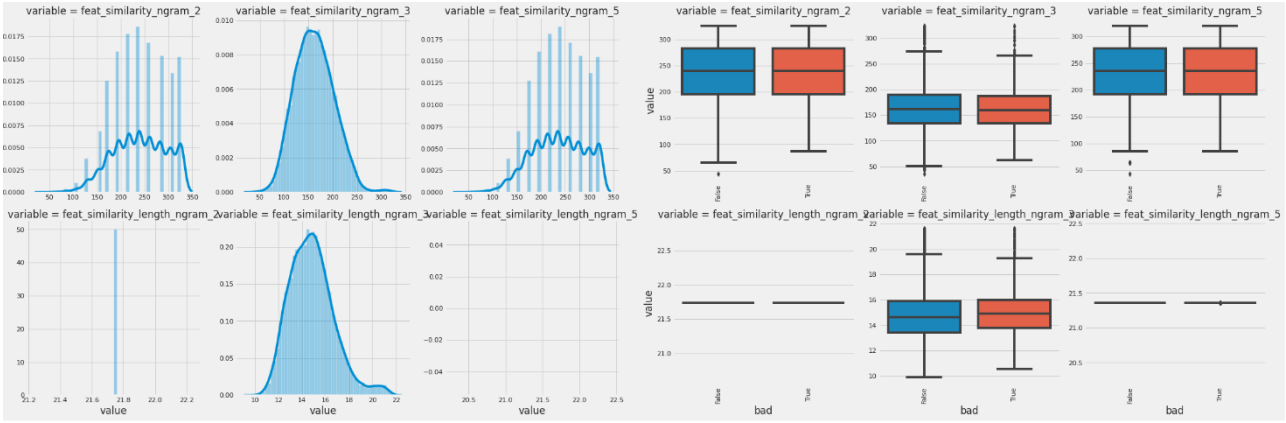

Fig. 22. Twitter dataset visualizations of entropy-based similarity

3.5.4. Specification-based features

Based on the specifics of the dataset, we are recreating convenience options as features for future models or usage of popular email service as a feature, in total we managed to retrieve 5 features for email and 2 features for the Twitter dataset. We can visualize count plots for different user classes and observe if there is any deviation based on the class.

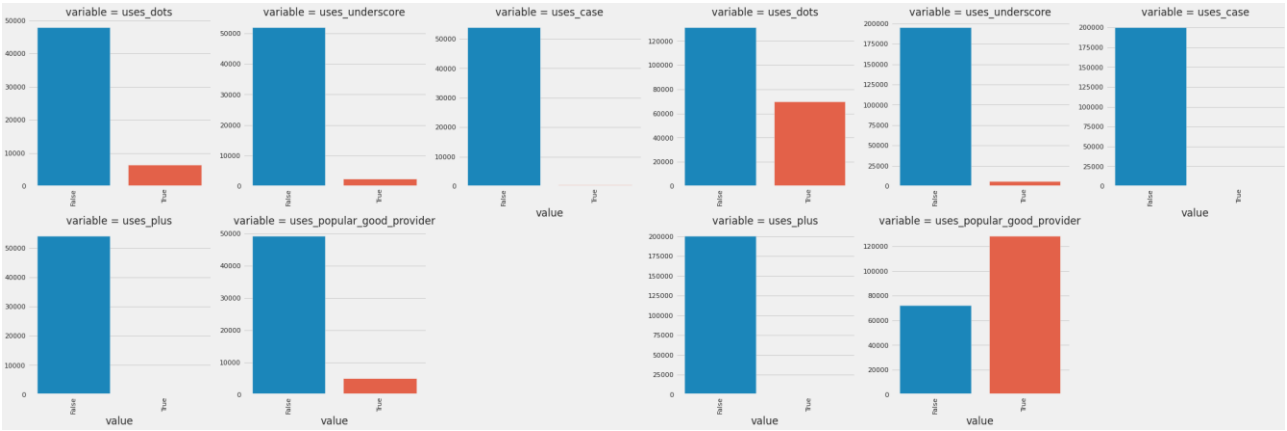

Fig. 23. Email data set, specification-based features. Left – bad users, right – good users.

For email dataset visualization of technical features (Figure 23) we can see that few features have different usage between good and bad usernames. For example, usage of dots or good email providers is more typical for good users, while malicious users do not use those.

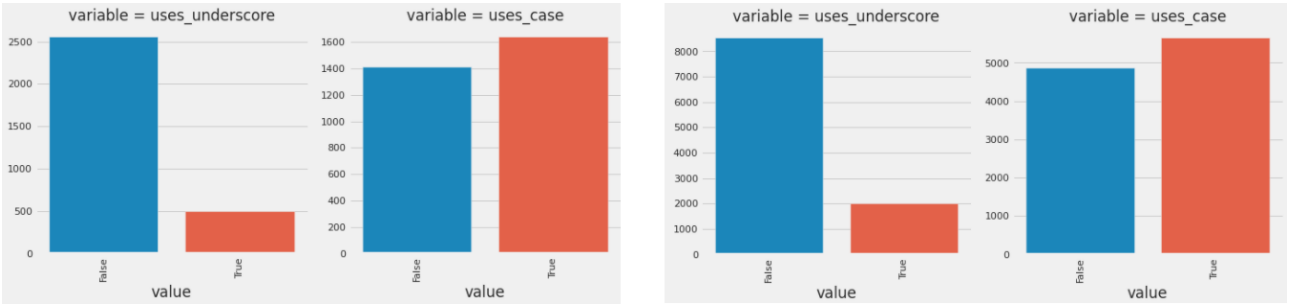

Fig. 24. Twitter data set, specification-based features. Left – bad users, right – good users.

For the Twitter dataset, there is no slight difference in those features (Figure 24), which concerns quite at this point in the research. However, with a full combination of features, there is a possible chance that machine learning algorithms will be able to observe more patterns comparing to our eye and catch them.

3.5.5. Linguistic-based features

For those features, we prepared different types of measurement – categorical as detected language by CLD ver 2/3, quantitative – number of vowels, the proportion of vowel, number of numbers (as there is no need for proportion being calculated, as this was done by the amount of top keyboards rows). To measure demographic bias, we will use language prediction level as input for the feature as well.

After we plot the boxplots by variable for emails (Figure 25), we can observe that several quantitative have distinct distribution cuts. That means, that good users prefer fewer digits in their names and bad users generally use more vowels, however, the proportion of them is slightly less, from which we can make a conclusion, that bad users are using longer email generally compared to the good user.

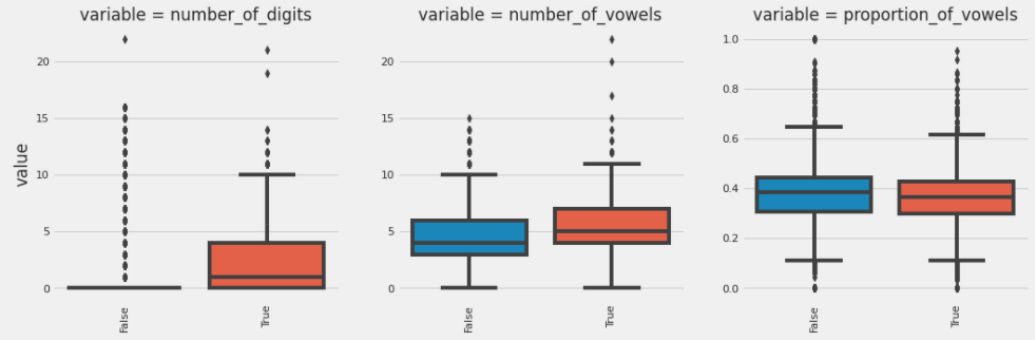

Fig. 25. Boxplot by the value of the linguistic features for emails

After that, we visualize count plots of all predictions of languages (Figure 26). We can see that there a lot of different languages detected in the dataset, however, one dominated all other languages, which is not surprising, as such methods work worse, when they are not exposed to the long strings.

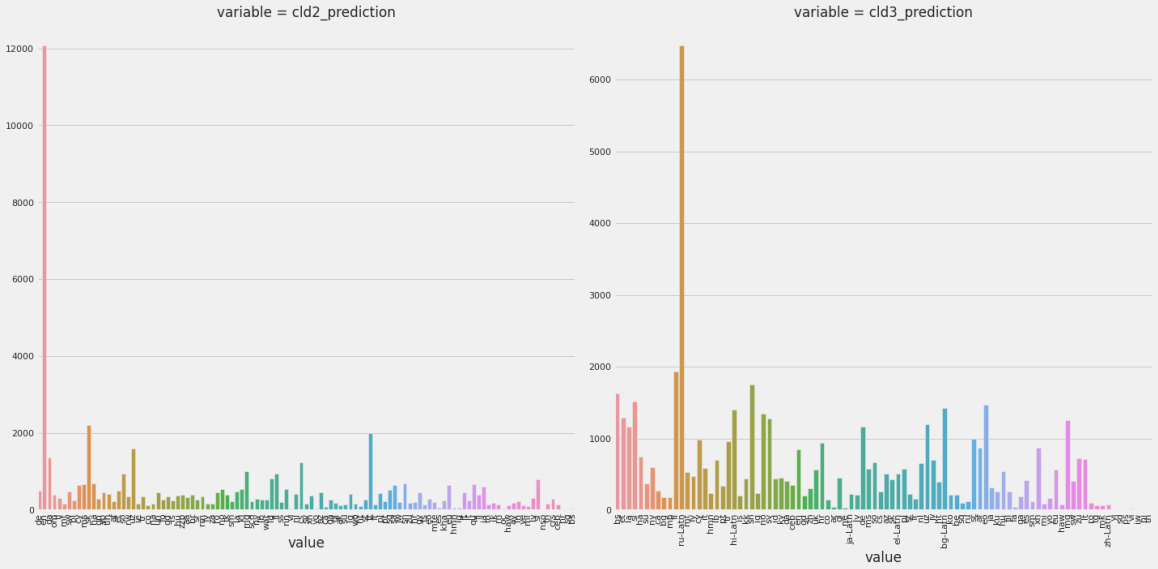

Fig. 26. Countplot of the language predictions for emails by Compact Language detector.

If the same visualizations for the Twitter username dataset are plotted – we can observe that there almost no difference between distributions of two classes (Figure 27), which is as it was mentioned earlier is quite concerning.

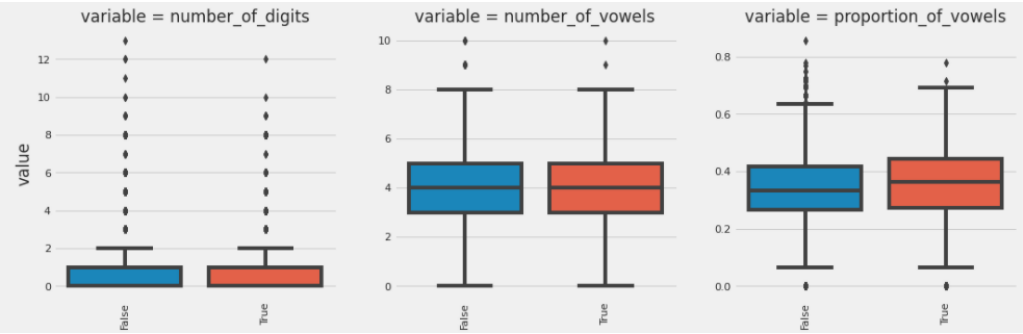

Fig. 27. Linguistic-based feature boxplot by class for Twitter dataset

Language detection histogram looks like with email datasets. These results we encoded, so it is possible to use more classification methods.

3.6. Model training

After the feature engineering model training can be started. In this research classification models at the beginning, separately will be trained with different feature sets, to measure how different characteristics represented by the features may impact model results. Those results will be presented in the result summary as a benchmark to track model improvements when all features are used. In total this will create 4 groups of models (2 groups for each dataset).

For both datasets, there are imbalanced class distributions. For the Twitter dataset, it is 10523 good entries and 3051 bad, which leads to 77.5:22.5 disbalance. For Email datasets, it is 201008 good emails and 54344 bad emails, which is 79:21 class inequality.

As mentioned earlier - some methods, such as assigning weights to the classes or limiting maximal delta, for better prediction of the imbalanced class will be used.

3.6.1. Username classification models using separate groups of features

Typing behavior-based features. During feature engineering, there were generated features, which represent typing behavior, movements of fingers, hands, usage of certain keystrokes or rows. To measure the impact of those features, first PCA decomposition is performed to check the variance of observations. In total there are 84 features, which means that 84 component decomposition can be performed, but such decomposition is redundant. For this feature set, 10 component decomposition is done, to check, how data is distributed after dimensionality reduction. As it is seen from the 1st and 2nd component graph (Figure 28), bad and good users do not differ much, however from the 2nd and 3rd component visualization (Figure 28), it seems, that some observations of fraudulent and good users start to form a separate cluster.

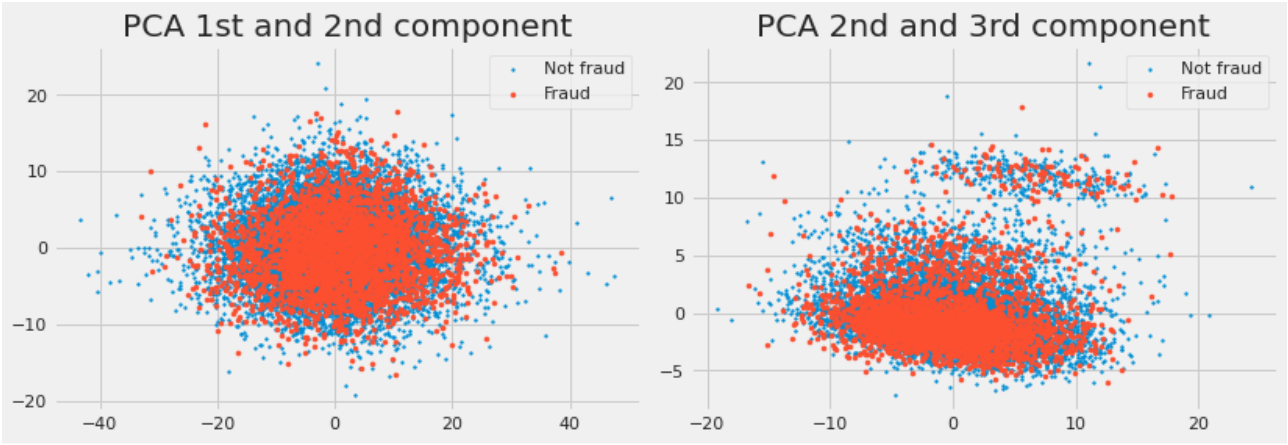

Fig. 28. Typing behavior-based feature PCA decomposition.

To classify usernames, during classification model training Linear Regression, Support Vector Machine, Random Forest and several variations of Gradient Boosting algorithms will be used. The dataset is randomly sampled in 80:20.

Table 7. Typing behaviors testing model results

Model	Score	Sensitivity	Specificity
Gradient Boosting	0.63277	0.944076	0.163636
LGBM dart 1500 est. with subsample	0.621647	0.987678	0.105785
LGBM dart 1000 est.	0.620874	0.988152	0.105785
XGBoost	0.617973	0.940758	0.173554
LGBM default classifier	0.617047	0.987678	0.100826
Random Forest	0.610759	0.908531	0.249587
Logistic Regression	0.605102	0.997156	0.052893
ADA Boosting	0.603123	0.998104	0.029752
Extra Tree	0.602441	0.859716	0.28595
SVM	0.595301	0.610427	0.512397
Decision Tree	0.587738	0.631754	0.494215

As it is seen, the best AUC results are achieved with gradient boosted models (Table 7). However, specificity is around 0.16, which means that only 16 percent of malicious users are detected using only typing behavior features.

After running through easyROC, it is seen (Table 8), that the best performing models do not have independent predictions as the *p*-value is above 0.05, that is why combining their outputs in the assembly may not give the increase in detection, however, averaging them may help to increase classification accuracy.

Table 8. easyROC multiple comparison. Twitter username, typing behavior best models.

Marker1 (I)	Marker2 (J)	AUC(I)	AUC(J)	|I - J|	SE(|I - J|)	z	p-value	p-value (adj.)
LGBM_dart_1000	LGBM_dart_1500	0.6209	0.6216	0.0008	0.0188	0.0411	0.9672	1
LGBM_dart_1000	LGBM.default_classifier	0.6209	0.617	0.0038	0.0188	0.2039	0.8384	1
LGBM_dart_1000	XGB	0.6209	0.618	0.0029	0.0188	0.1542	0.8775	1
LGBM_dart_1000	GB	0.6209	0.618	0.0029	0.0188	0.1542	0.8775	1
LGBM_dart_1500	LGBM.default_classifier	0.6216	0.617	0.0046	0.0188	0.2442	0.807	1
LGBM_dart_1500	XGB	0.6216	0.618	0.0037	0.0189	0.1946	0.8457	1
LGBM_dart_1500	GB	0.6216	0.618	0.0037	0.0189	0.1946	0.8457	1
LGBM.default_classifier	XGB	0.617	0.618	0.0009	0.0189	0.0491	0.9608	1

LGBM.default_classifier	GB	0.617	0.618	0.0009	0.0189	0.0491	0.9608	1
XGB	GB	0.618	0.618	0	0.0189	0	1	1

Information-based features. The same routine which was made with typing-behavior features will be done with information-based features. However, as there are only 7 features, PCA decomposition was limited to 4 components. As we can see from visualization of PCA (Figure 29), that most of the malicious observations are grouped with good observations, which can tell, that left alone such model will not give any positive results.

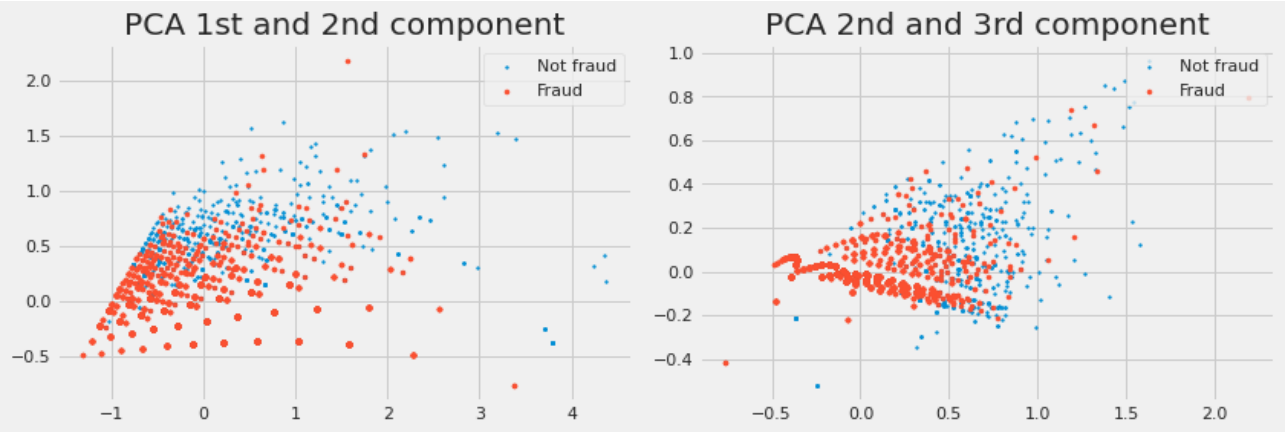

Fig. 29. Information-based features PCA

After the training process and hyperparameter optimization, there are only two types of created models. Those two types are the following: classify everything as a good class, because of that they have sensitivity 1 and specificity 0 or have both Sensitivity and Specificity around 0.5 (Table 9). Both types are bad examples of classification models that is why it is possible to conclude, that based on the information-based features alone, it is impossible to determine if the user is fraudulent or not. This repeats trends observed during the Exploratory Data Analysis of generated features. As most of those model outputs provide unstable results (according to the AUC definition in Section 2.5) and cannot correctly classify, multiple comparisons with EasyROC will not be made.

Table 9. Information-based features testing model results.

Model	Score	Sensitivity	Specificity
LGBM default classifier	0.579277	0.999526	0
ADA Boosting	0.579277	0.999526	0
Extra Tree	0.576154	0.577725	0.532231
LGBM dart 1000	0.573725	1	0
LGBM dart 1500 est. with subsample	0.573216	1	0
XGBoost	0.572276	0.996209	0.004959
Random Forest	0.571997	0.555924	0.563636
Gradient Boosting	0.567551	0.994313	0.006612

Decision Tree	0.563056	0.53981	0.570248
SVM	0.559277	0.411848	0.634711
Logistic Regression	0.558455	1	0

Similarity with information entropy. The next model group will be trained with the feature set, representing similarities in the usernames and encoding repeating patterns. As with previous model training, we will perform PCA decomposition (Figure 30). As we have only 6 features, we will use 4 component PCA decomposition and visualize scatter plots of the first components. It is quite hard to interpret such decomposition results, but it seems, that observations (usernames) formed clusters.

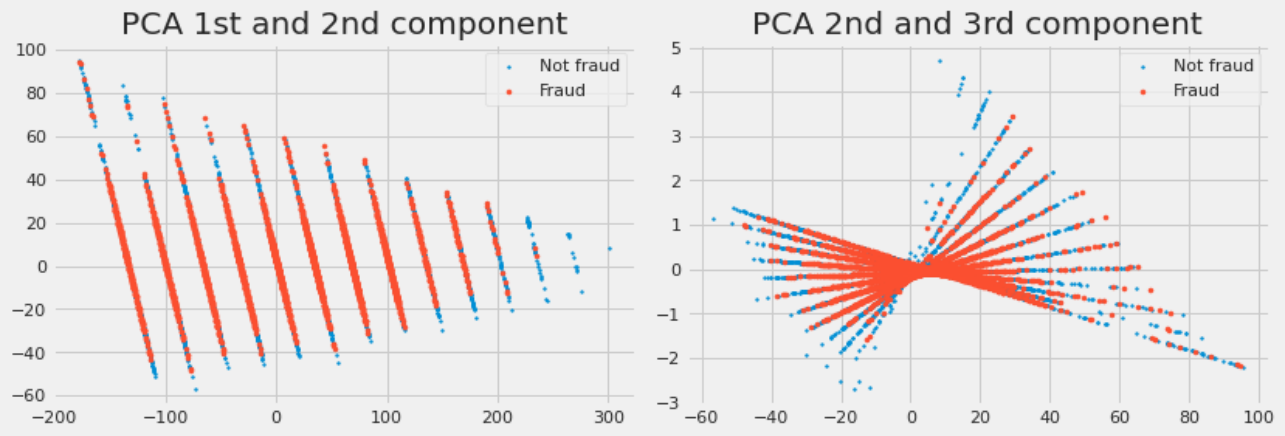

Fig. 30. PCA decomposition using similarity-based features.

However, during the training step, a similar trend as with information-based features was observed. Changing the number of estimators or complexities of the model ensembles did not help to reach results where classification was significantly better than randomly assigning classes for observation. For this reason, as previously we will use the AUC score of the model, which did not classify all usernames as a good one (Table 10). Same as during the previous feature groups, when the models did not provide a stable classifier, there are no multiple comparisons made using easyROC.

Table 10. Similarity features testing model scores.

Model	Score	Sensitivity	Specificity
LGBM default classifier	0.585602	1	0
LGBM dart 1500 est. with subsample	0.585375	0.999526	0.001653
Extra Tree	0.583242	0.620379	0.492562
ADA Boosting	0.579277	0.999526	0
LGBM dart 1000	0.573757	0.998104	0.003306
Random Forest	0.564428	0.656872	0.404959
SVM	0.564405	0.444076	0.629752
Decision Tree	0.561342	0.557346	0.523967
Logistic Regression	0.557811	1	0
XGBoost	0.54261	0.967773	0.036364

Gradient Boosting	0.536934	0.964455	0.044628

Specification-based features. For Twitter, only 3 features were calculated in this group, and all of them have Boolean values (True or False values), which is why PCA decomposition is not performed. During training, it is observed, that there exists a similar inability to find a function, which could classify if the username is malicious or not with this set of features for Twitter usernames (best AUC is with Extra Tree – 0.58). That is the reason why multiple comparisons of model outputs were not performed.

Linguistic-based features. As those features mostly consist of categorical data, PCA decomposition was not done. For training, all the categorical data was encoded with dummies, as not all proposed algorithms can deal with categorical data. After encoding categorical features, in total there are 208 features. Based on the training results, it can be noticed that by using linguistic-based features alone it is possible to build a classifier. The results are similar to keyboard behavior features, while the best-performing models remain based on Gradient Boosting optimization (Table 11).

Table 11. Linguistic-based features testing model results.

Model	Score	Sensitivity	Specificity
LGBM dart 1000	0.606902	0.981043	0.110744
LGBM default classifier	0.604035	0.982464	0.112397
LGBM dart 1500 est. with subsample	0.600913	0.981991	0.107438
Random Forest	0.597742	0.80237	0.330579
XGBoost	0.596317	0.959716	0.168595
Decision Tree	0.596265	0.692417	0.439669
Gradient Boosting	0.590098	0.957346	0.155372
ADA Boosting	0.585931	0.99763	0.006612
Logistic Regression	0.585375	0.995261	0.009917
SVM	0.579478	0.633649	0.495868

After performing multiple comparisons of model results using easyROC it is observed that there are no significant differences between model decisions if they all use the same linguistic features. This can be confirmed in Table 12, as there is no pair of comparisons, where the *p*-value is less than 0.05.

Table 12. easyROC multiple comparison. Twitter username, linguistic best models.

Marker1 (I)	Marker2 (J)	AUC(I)	AUC(J)	|I - J|	SE(|I - J|)	z	p-value	p-value (adj.)
LGBM_dart_1000	LGBM_dart_1500	0.6069	0.6009	0.006	0.0193	0.3102	0.7564	1
LGBM_dart_1000	LGBM.default_classifier	0.6069	0.604	0.0029	0.0193	0.1483	0.8821	1
LGBM_dart_1000	XGB	0.6069	0.5963	0.0106	0.0195	0.5429	0.5872	1

LGBM_dart_1000	GB	0.6069	0.5963	0.0106	0.0195	0.5429	0.5872	1
LGBM_dart_1500	LGBM.default_classifier	0.6009	0.604	0.0031	0.0194	0.1611	0.872	1
LGBM_dart_1500	XGB	0.6009	0.5963	0.0046	0.0195	0.2352	0.8141	1
LGBM_dart_1500	GB	0.6009	0.5963	0.0046	0.0195	0.2352	0.8141	1
LGBM.default_classifier	XGB	0.604	0.5963	0.0077	0.0196	0.3944	0.6933	1
LGBM.default_classifier	GB	0.604	0.5963	0.0077	0.0196	0.3944	0.6933	1
XGB	GB	0.5963	0.5963	0	0.0197	0	1	1

Subsection summary. During the training separately by features, only two feature sets showed the efficiency of being used for classification alone – Typing behavior and Linguistic based features. For each of the feature sets, the best classifying models were LGBM, main differences are hyperparameters used in training, one was using classic classifier function, and another one – DART classifier. In the next subsection, we will try to train models using all features together.

Additionally, before using all features as classifiers, averaged model using outputs of the best models of Typing Behavior and Linguistic based was introduced. In the averaged model-new probability was the average of Typing Behavior and Linguistic model. This way, a slight improvement of the following metrics was achieved: AUC slightly increased to 0.6208742, sensitivity – 0.99575 and specificity – 0.09752 at 0.5 threshold. According to the best cut-off by Youden methodology, when the cost classes are equal, the best threshold would be 0.336, this would give sensitivity – 0.94502 and specificity – 0.22.

3.6.2. Username evaluation final models with all features

For this final model in training, all calculated features are used. PCA decomposition is performed using non-categorical data. No matter if 20 or more component number is picked, first 10 explain 99% of the variance in data. As it can be observed, there are several clusters of observations if scatterplots of observations are plotted by PCA components (Figure 31).

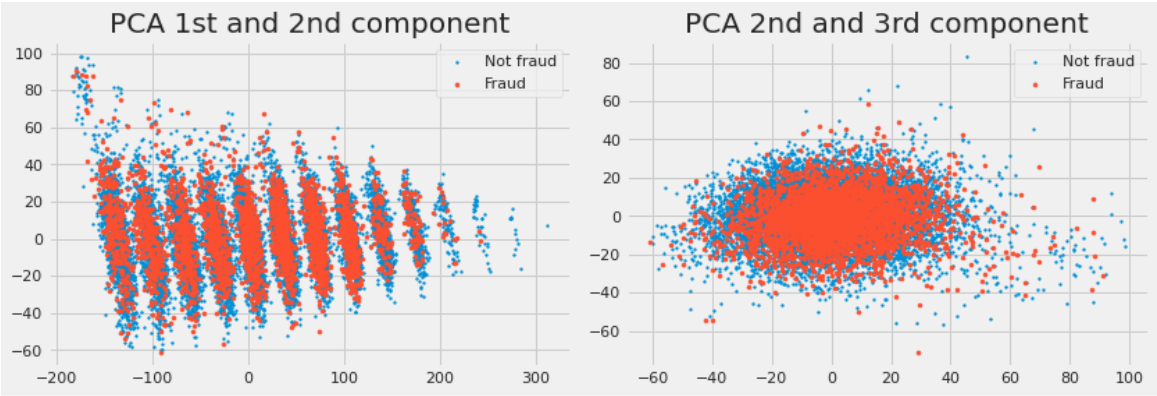

Fig. 31. PCA decomposition plots for 1-3 components

PCA between 2nd and 3rd components is quite similar to PCA between 1st and 2nd components for the data, where only Typing Behavior-based features were used. After performing training and testing of models - improvements can be observed for all model performances with almost all algorithms (Table 13).

Table 13. Final model testing results with all features.

Model	Score	Sensitivity	Specificity
LGBM dart 1000 est.	0.648391	0.991943	0.155372
Gradient Boosting	0.645552	0.949763	0.214876
LGBM dart 1500 est. with subsample	0.645252	0.991943	0.152066
LGBM default classifier	0.6407	0.990995	0.157025
Random Forest	0.638883	0.9109	0.26281
XGBoost	0.628646	0.958768	0.196694
ADA Boosting	0.624022333	0.994313	0.069421
Extra Tree	0.616837	0.824645	0.332231
Logistic Regression	0.603235	0.996682	0.026446
Decision Tree	0.601852	0.640758	0.46281
SVM	0.575763	0.408057	0.661157

For example, the best results show, that models improved not only Sensitivity but Specificity as well. In theory, with any of the top 2 models, it is possible to detect more than 15% of malicious users only based on their username. As well, False Positive is a little less than 1%. After plotting the best 5 performing models, it is noticeable that their performance is quite similar (Figure 32).

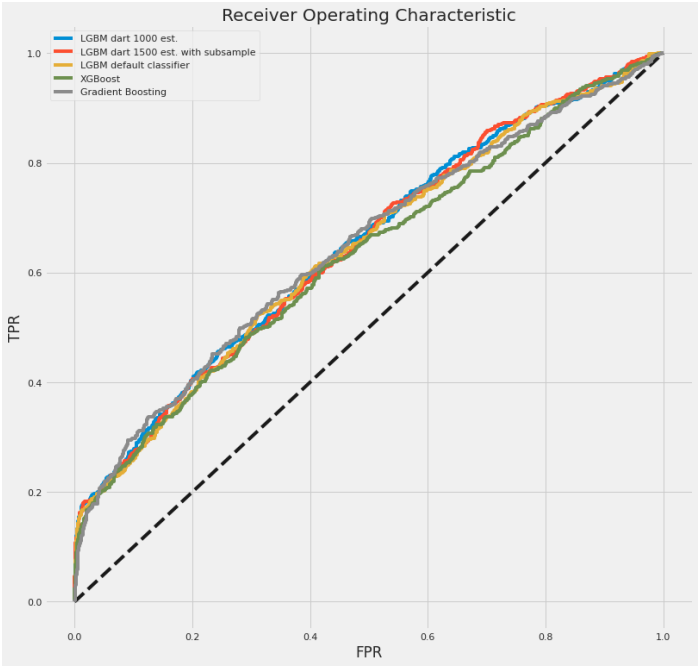

Fig. 32. Receiver Operating Characteristic for Twitter username TOP5 models

After multiple pair comparisons of the best performing models with easyROC, there is no model pair, there exists a significant statistical difference. That is why it is possible to tell, that all models identify similar malicious users.

Table 14. easyROC comparison. Twitter username, linguistic best models.

Marker1 (I)	Marker2 (J)	AUC(I)	AUC(J)	|I - J|	SE(|I - J|)	z	p-value	p-value (adj.)
LGBM_dart_1000	LGBM_dart_1500	0.6484	0.6453	0.0031	0.0186	0.1692	0.8657	1
LGBM_dart_1000	LGBM.default_classifier	0.6484	0.6407	0.0077	0.0187	0.4121	0.6803	1
LGBM_dart_1000	XGB	0.6484	0.6286	0.0197	0.0188	1.0487	0.2943	1
LGBM_dart_1000	GB	0.6484	0.6286	0.0197	0.0188	1.0487	0.2943	1
LGBM_dart_1500	LGBM.default_classifier	0.6453	0.6407	0.0046	0.0186	0.2446	0.8068	1
LGBM_dart_1500	XGB	0.6453	0.6286	0.0166	0.0188	0.8844	0.3765	1
LGBM_dart_1500	GB	0.6453	0.6286	0.0166	0.0188	0.8844	0.3765	1
LGBM.default_classifier	XGB	0.6407	0.6286	0.0121	0.0189	0.6382	0.5234	1
LGBM.default_classifier	GB	0.6407	0.6286	0.0121	0.0189	0.6382	0.5234	1
XGB	GB	0.6286	0.6286	0	0.019	0	1	1

According to the feature importance for top model (Appendix 3): the *LGBM dart 1000 est.* – the most important features consist of different types of features (however the majority of TOP10 – typing features). 2 in the top 5 features are from the feature sets which could not build a model on their own (Table 15).

Table 15. TOP 10 feature importance of *LGBM dart 1000 est.* model for the Twitter usernames

Feature name	F Score	Feature name	F Score
feat_similarity_ length_ngram_3	1195	feat_similarity_ ngram_3	619
feat_linguistic_ proportion_of_vowels	1160	feat_typing_ DVORAK_log_euc_average	589
feat_linguistic_ cld3_proba	944	feat_typing_ dvorak_bot_row_percentage	585
feat_technical_ uses_case	847	feat_typing_ qwerty_num_row_percentage	556
feat_typing_ dvorak_same_row_proportion	665	feat_typing_ qwerty_same_row_proportion	496

The confusion matrix fully mirrors calculated metrics – 94 malicious users detected, while 17 good users were identified wrongly as malicious (Figure 33).

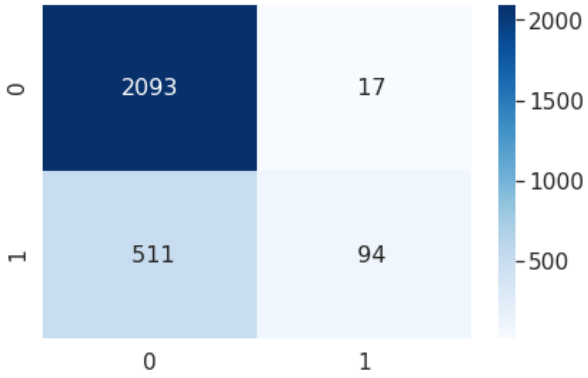

Fig. 33. Confusion matrix of the best performing model with all features.

Model comparison and discussion will be in the result summary section.

3.6.3. Email evaluation models using a separate group of features

Typing behavior-based features. In the beginning, 10 component PCA decomposition is performed. Although PCA 1st and 2nd scatterplots (Figure 34) do not seem to separate observations in different clusters, however, they explain over 95% of the variation. This confirms observations made during EDA for those features, that there are some features, where the difference between populations of two classes is noticeable without statistical tests.

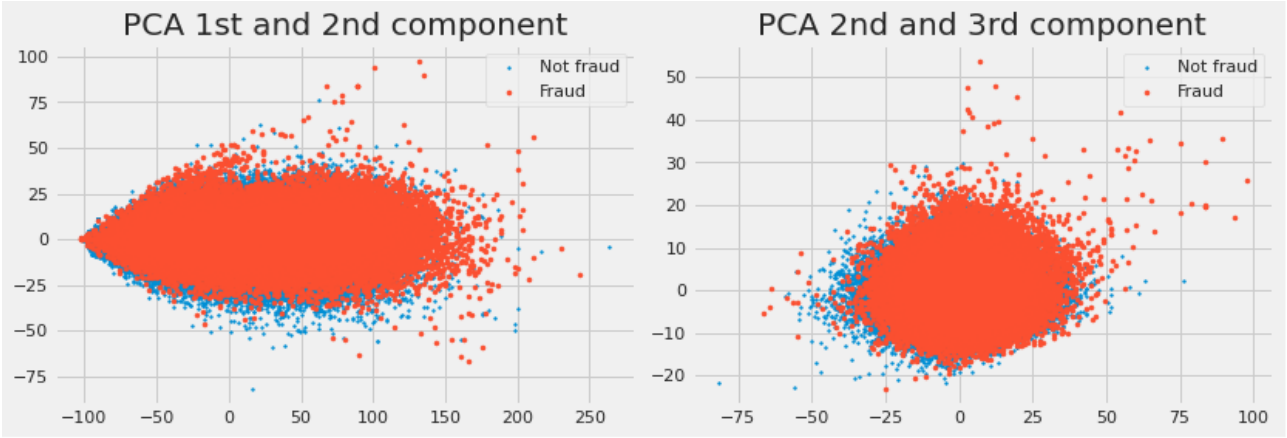

Fig. 34. PCA decomposition for typing features for email dataset.

After the training with algorithms used previously, we can see, that those models are quite efficient predicting, not only for Twitter usernames but emails as well. As it was noticed, that AdaBoost and SVM have suboptimal results and it is very hard to stop overfitting (while the training process takes a long time) – they were not used for email dataset.

In the case of usage typing-behavior features only it is already possible to identify more than 50% of malicious users' email addresses with a False Positive rate, less than 3% (Table 16), which is a massive improvement comparing with the Twitter use-case.

Table 16. Email evaluation model based on typing-behavior features.

Model	Score	Sensitivity	Specificity
XGBoost	0.898516	0.974737	0.560871
Gradient Boosting	0.889331	0.97389	0.546511
LGBM dart 1000 est.	0.876058	0.981364	0.489619
LGBM default classifier	0.874902	0.978325	0.498125
LGBM dart 1500 est. with subsample	0.873472	0.981813	0.482027
Random Forest	0.869071	0.922791	0.626086
Extra Tree	0.848706	0.925432	0.59261
Decision Tree	0.82186	0.809408	0.674014
Logistic Regression	0.813834	0.969156	0.422208

After performing model comparison (Table 17), it can be seen, that the following models give statistically independent predictions and could be used to form a new assemble of models. Specifically, LGBM models are not statistically independent between themselves, same as XGB Gradient Boosting (*p*-value is greater than 0.05), however all LightGBM trained models have significant statistical difference with XGB and Gradient Boosting in outputs (*p*-value is less than 0.05).

Table 17. easyROC comparison. Email address, typing behavior-based features best models.

Marker1 (I)	Marker2 (J)	AUC(I)	AUC(J)	|I - J|	SE(|I - J|)	z	p-value	p-value (adj.)
XGB	GB	0.8985	0.8985	0	0.0025	0	1	1
LGBM_dart_1000	LGBM.default_classifier	0.8761	0.8749	0.0012	0.0028	0.413	0.6796	1
LGBM_dart_1500	LGBM.default_classifier	0.8735	0.8749	0.0014	0.0028	0.5091	0.6107	1
LGBM_dart_1000	LGBM_dart_1500	0.8761	0.8735	0.0026	0.0028	0.9187	0.3582	1
LGBM_dart_1000	XGB	0.8761	0.8985	0.0225	0.0026	8.4971	0	0
LGBM_dart_1000	GB	0.8761	0.8985	0.0225	0.0026	8.4971	0	0
LGBM_dart_1500	XGB	0.8735	0.8985	0.025	0.0027	9.4272	0	0
LGBM_dart_1500	GB	0.8735	0.8985	0.025	0.0027	9.4272	0	0

LGBM.default_classifier	XGB	0.8749	0.8985	0.0236	0.0026	8.9499	0	0
LGBM.default_classifier	GB	0.8749	0.8985	0.0236	0.0026	8.9499	0	0

Information-based features. As there are fewer features, comparing to typing behavior-based, we will reduce components in PCA analysis down to 4. As it is can be seen from the first 3 PCA components, malicious users start to form a separate cluster, which can result in successful classification (Figure 35).

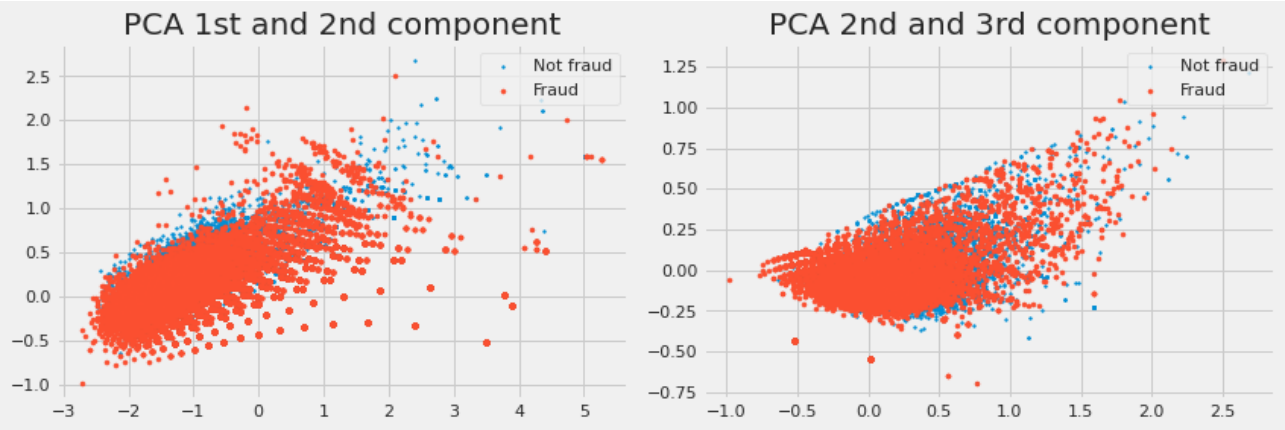

Fig. 35. PCA decomposition of the email dataset with information-based features.

After training with selected classifying algorithms, it can be noted that the same tendency which was observed with the Twitter username dataset does not reproduce and it is possible to classify emails by that feature set. Though it is not as effective as the identification with typing behavior features, however, it still catches 18% of malicious users with falsely identifying good users as bad ones with a 2.4% rate (Table 18).

Table 18. Email evaluation model based on information features.

Model	Score	Sensitivity	Specificity
XGBoost	0.697749	0.976855	0.189427
Random Forest	0.697664	0.741492	0.55273
LGBM default classifier	0.696879	0.978101	0.186865
Extra Tree	0.69672	0.763715	0.527577
Gradient Boosting	0.696513	0.978873	0.182018
LGBM dart 1000 est.	0.696374	0.977951	0.185311
LGBM dart 1500 est. with subsample	0.695696	0.978101	0.185676
Decision Tree	0.692575	0.733071	0.555108
Logistic Regression	0.665379	0.995167	0.073996

After performing multiple comparisons between best performing models, which sensitivity was over 0.9 at 0.5 cut-out (Table 19), it was observed, that no models have statistically significant different outputs (all *p*-values are more than 0.05).

Table 19. easyROC comparison. Email address, information-based features best models.

Marker1 (I)	Marker2 (J)	AUC(I)	AUC(J)	|I - J|	SE(|I - J|)	z	p-value	p-value (adj.)
LGBM_dart_1000	LGBM_dart_1500	0.6964	0.6957	0.0007	0.0042	0.1596	0.8732	1
LGBM_dart_1000	LGBM.default_classifier	0.6964	0.6969	0.0005	0.0042	0.1194	0.905	1
LGBM_dart_1000	XGB	0.6964	0.6977	0.0014	0.0042	0.3243	0.7457	1
LGBM_dart_1000	GB	0.6964	0.6977	0.0014	0.0042	0.3243	0.7457	1
LGBM_dart_1500	LGBM.default_classifier	0.6957	0.6969	0.0012	0.0042	0.2789	0.7803	1
LGBM_dart_1500	XGB	0.6957	0.6977	0.0021	0.0042	0.4835	0.6287	1
LGBM_dart_1500	GB	0.6957	0.6977	0.0021	0.0042	0.4835	0.6287	1
LGBM.default_classifier	XGB	0.6969	0.6977	0.0009	0.0042	0.2051	0.8375	1
LGBM.default_classifier	GB	0.6969	0.6977	0.0009	0.0042	0.2051	0.8375	1
XGB	GB	0.6977	0.6977	0	0.0042	0	1	1

Similarity with information entropy. Same as with the previous feature set, decomposition is performed to see the variance of the data and if the classes are visually separated. Based on representation it is possible to tell that classes are getting separated, which could mean, that the classification can be more successful comparing to the experience with Twitter datasets (Figure 36).

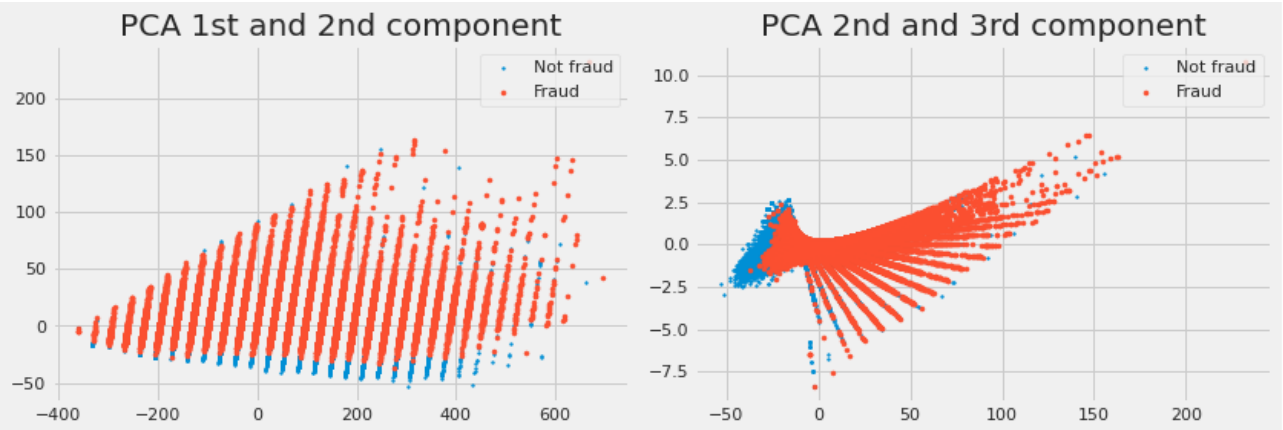

Fig. 36. PCA decomposition with similarity features

Opposite to the Twitter case, this feature set handles information about fraudulent users, and it is possible to identify malicious users with moderate accuracy – achieving a 48% detection rate of malicious users and accurately classifying good users with a Sensitivity of 0.97 (Table 20).

Table 20. Email evaluation model based on similarity features.

Model	Score	Sensitivity	Specificity
XGBoost	0.856026	0.973766	0.484131
Gradient Boosting	0.850647	0.972146	0.481021
LGBM default classifier	0.837928	0.978101	0.43666
LGBM dart 1000 est.	0.836382	0.978997	0.432635
LGBM dart 1500 est. with subsample	0.835178	0.97982	0.430531
Random Forest	0.81624	0.797997	0.652977
Logistic Regression	0.805666	0.968932	0.403915
Decision Tree	0.803053	0.869575	0.564255
Extra Tree	0.79878	0.729334	0.693131

After performing multiple comparisons with the easyROC tool, it is possible to see, that all models return results, which are not significantly different as the *p*-value is higher than 0.05 (Table 21).

Table 21. easyROC comparison. Email address, similarity features best models.

Marker1 (I)	Marker2 (J)	AUC(I)	AUC(J)	|I - J|	SE(|I - J|)	z	p-value	p-value (adj.)
LGBM_dart_1000	LGBM_dart_1500	0.8082	0.8075	0.0007	0.0033	0.2162	0.8288	1
LGBM_dart_1000	LGBM.default_classifier	0.8082	0.8086	0.0005	0.0032	0.1414	0.8875	1
LGBM_dart_1000	XGB	0.8082	0.8069	0.0012	0.0033	0.3803	0.7037	1
LGBM_dart_1000	GB	0.8082	0.8069	0.0012	0.0033	0.3803	0.7037	1
LGBM_dart_1500	LGBM.default_classifier	0.8075	0.8086	0.0012	0.0033	0.3576	0.7207	1
LGBM_dart_1500	XGB	0.8075	0.8069	0.0005	0.0033	0.1638	0.8699	1
LGBM_dart_1500	GB	0.8075	0.8069	0.0005	0.0033	0.1638	0.8699	1
LGBM.default_classifier	XGB	0.8086	0.8069	0.0017	0.0033	0.5218	0.6018	1
LGBM.default_classifier	GB	0.8086	0.8069	0.0017	0.0033	0.5218	0.6018	1
XGB	GB	0.8069	0.8069	0	0.0033	0	1	1

Technical-based features. As all the features are categorical, PCA decomposition is skipped and model training is performed. However, after looking into results, the same trend reoccurs, which was noticed with Twitter usernames. The best model iteration is along with 0.61 AUC and TPR of 0.34 and TNR of 0.88.

Linguistic-based features. As most of the features are categorical, PCA analysis is not made and model training begins after categorical data reencoding into dummy variables. After displaying model metrics – it is seen that most models are effective to detect malicious users (Table 22). Metrics are slightly lower comparing to the typing behavior features. However, it still possible to use such a classifier, as it detects around 48% of malicious emails, with a False Positive rate below 3%, which for a classifier that works only with a limited amount of information is perfect.

Table 22. Email evaluation model based on linguistic features.

Model	Score	Sensitivity	Specificity
XGBoost	0.856026	0.973766	0.484131
Gradient Boosting	0.850647	0.972146	0.481021
LGBM default classifier	0.837928	0.978101	0.43666
LGBM dart 1000 est.	0.836382	0.978997	0.432635
LGBM dart 1500 est. with subsample	0.835178	0.97982	0.430531
Random Forest	0.81624	0.797997	0.652977
Logistic Regression	0.805666	0.968932	0.403915
Decision Tree	0.803053	0.869575	0.564255
Extra Tree	0.79878	0.729334	0.693131

After performing multiple comparisons of model results with the help of the EasyROC tool, it is observed, that most of the models provide independent results as the p-value is less than 0.05 (Table 23). The only models which provide similar results are models using similar Tree building algorithms – LightGBM.

Table 23. easyROC comparison. Email address, similarity features best models.

Marker1 (I)	Marker2 (J)	AUC(I)	AUC(J)	|I - J|	SE(|I - J|)	z	p-value	p-value (adj.)
LGBM_dart_1000	LGBM_dart_1500	0.8364	0.8352	0.0012	0.0032	0.3763	0.7067	1
LGBM_dart_1000	LGBM.default_classifier	0.8364	0.8379	0.0015	0.0032	0.4855	0.6273	1
LGBM_dart_1000	XGB	0.8364	0.856	0.0196	0.0031	6.364	0	0
LGBM_dart_1000	GB	0.8364	0.856	0.0196	0.0031	6.364	0	0
LGBM_dart_1500	LGBM.default_classifier	0.8352	0.8379	0.0027	0.0032	0.8614	0.389	1

LGBM_dart _1500	XGB	0.8352	0.856	0.0208	0.0031	6.7336	0	0
LGBM_dart _1500	GB	0.8352	0.856	0.0208	0.0031	6.7336	0	0
LGBM.defa ult_classifier	XGB	0.8379	0.856	0.0181	0.0031	5.8782	0	0
LGBM.defa ult_classifier	GB	0.8379	0.856	0.0181	0.0031	5.8782	0	0
XGB	GB	0.856	0.856	0	0.003	0	1	1

Model comparison between features will be performed in the Result summaries section.

3.6.4. Email evaluation models with all features

In overall separate feature groups with email datasets performed better compared to the datasets consisting of Twitter usernames, that is why it is expected that the email dataset will perform better with all features. After performing a 40 component PCA decomposition it can be seen that all calculated features explain the variance of emails (Figure 37) as the first 2 components describe almost 98% of the variance.

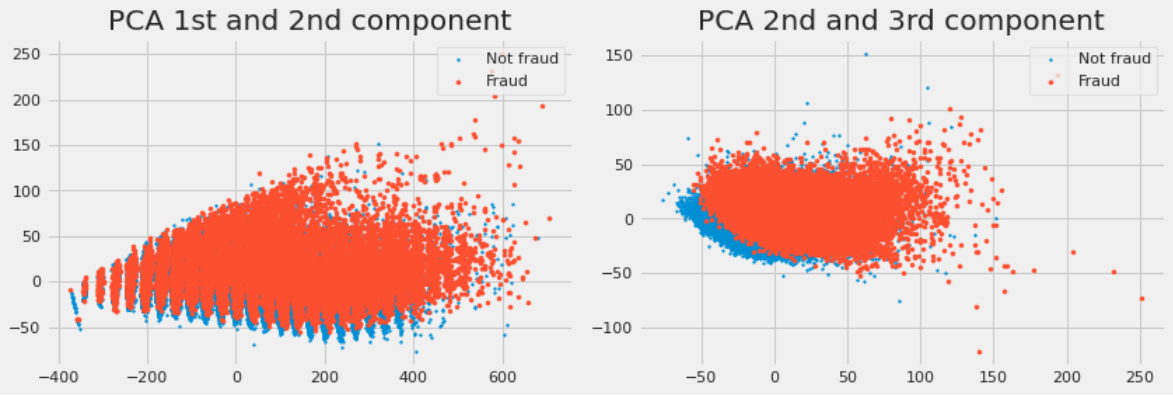

Fig. 37. PCA decomposition projections for email dataset with all generated features.

After running model verification following metrics displayed in Table 24 is achieved. It important to notice, that sort of models improved and while maintaining a false positive rate below 3%, it is possible to achieve TPR over 70%, which can be considered as very good results, according to the initial information amount – email address string. The best performing algorithms are algorithms using gradient boosting for the creation of an ensemble of weak classifiers. Logistic regression produces quite good results as well, however, the detection success rate is lower, and the false alarm rate for this algorithm is higher compared to the best model.

Table 24. Email evaluation model based on all features.

Model	Score	Sensitivity	Specificity
XGBoost	0.953453	0.970851	0.701088
Gradient Boosting	0.948377	0.967412	0.692216
LGBM dart 1000 est.	0.943442	0.97825	0.623525

LGBM dart 1500 est. with subsample	0.942523	0.977453	0.62133
LGBM default classifier	0.940654	0.974912	0.62947
Random Forest	0.934874	0.907843	0.763926
Extra Tree	0.927029	0.888435	0.773713
Decision Tree	0.911661	0.800787	0.836184
Logistic Regression	0.89322	0.957895	0.569377

Overall, training models with all features improve the quality of the model, however, there might be a possibility that training separately and using the ensemble model method for separate feature-based classifiers could improve overall performance more.

Multiple comparisons of the model outputs with the easyROC tool returned, that almost all model algorithms except for the LightGBM algorithm built have independent model results, as the *p*-value is less than 0.05.

Table 25. easyROC comparison. Email address, similarity features best models.

Marker1 (I)	**Marker2 (J)**	**AUC(I)**	**AUC(J)**	**|I - J|**	**SE(|I - J|)**	**z**	**p-value**	**p-value (adj.)**
LGBM_dart_1000	LGBM_dart_1500	0.9434	0.9425	0.0009	0.0015	0.5928	0.5533	1
LGBM_dart_1000	LGBM.default_classifier	0.9434	0.9407	0.0028	0.0016	1.7762	0.0757	0.757
LGBM_dart_1000	XGB	0.9434	0.9535	0.01	0.0015	6.8352	0	0
LGBM_dart_1000	GB	0.9434	0.9535	0.01	0.0015	6.8352	0	0
LGBM_dart_1500	LGBM.default_classifier	0.9425	0.9407	0.0019	0.0016	1.1858	0.2357	1
LGBM_dart_1500	XGB	0.9425	0.9535	0.0109	0.0015	7.424	0	0
LGBM_dart_1500	GB	0.9425	0.9535	0.0109	0.0015	7.424	0	0
LGBM.default_classifier	XGB	0.9407	0.9535	0.0128	0.0015	8.568	0	0
LGBM.default_classifier	GB	0.9407	0.9535	0.0128	0.0015	8.568	0	0
XGB	GB	0.9535	0.9535	0	0.0014	0	1	1

From the ROC curve, it is noticeable that most of the models perform similarly (Figure 38). However, the XGBoost curve is a little above the other, which is reflected in the AUC score.

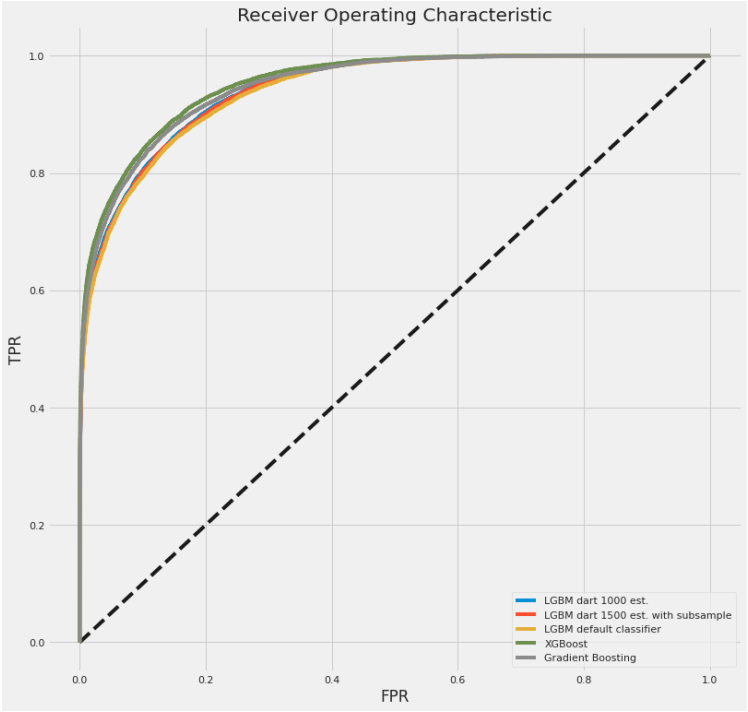

Fig. 38. ROC curve for TOP5 classifiers using all features on email dataset.

After plotting the feature importance plot, we observed that the most important features by F score are quite diverse and in the top 5 following groups are met: linguistic, technical, similarity, and typing.

Table 26. TOP 10 XGBoost feature by feature importance.

feat_technical_uses_popular_good_provider 0.1350481
feat_cld3_prediction__ru-Latn 0.030374104
feat_similarity_ngram_3 0.029546868
feat_typing_dvorak_num_row_percentage 0.02811064
feat_cld2_prediction__lt 0.027470104
feat_technical_uses_dots 0.021178149
feat_typing_DVORAK_manh_total 0.020000456
feat_typing_dvorak_right_hand_b 0.015411519
feat_cld3_prediction__zh-Latn 0.015018477
feat_typing_dvorak_bot_row_amount 0.012904751

As it can be observed from the Confusion Matrix, the model is capable to identify 7665 out of 10933 malicious users during classification, with only 1170 good users identified wrong (Figure 39).

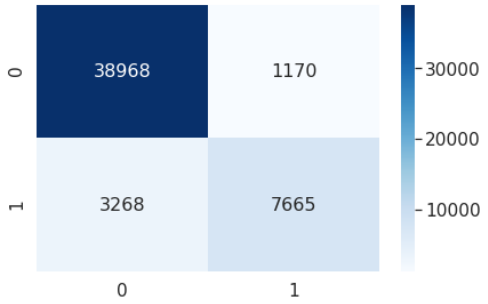

Fig. 39. Best email classification model‘s Confusion Matrix.

3.7. Result summaries and future research

In this research, we managed to create various features describing different types of information about the user, which could hint at the possibility of the user being a good one or a malicious one. Two different datasets were used for this purpose – email and Twitter username.

For each data set after feature calculation, exploratory data analysis was performed. During it, it became clear, that the email dataset has slightly different characteristics between classes, while the Twitter dataset has more similar population parameters for both classes. This could be reasoned because emails have less strict syntax restrictions, so malicious users do not need to get so creative as Twitter one, thus making them less like good users.

During training with the features calculated for Twitter, we created benchmarks and calculated for each feature set possible best model parameters, later to compare with the model, which would evaluate using all parameters as input for classification.

Table 27. Twitter dataset username classification model performance

Model type	Model name	AUC	Sensitivity	Specificity
Typing-behavior	Gradient Boosting	0.63277	0.944076	0.163636
Information-based	Extra Tree	0.567587	0.577725	0.532231
Similarity (information)	Extra Tree	0.567587	0.577725	0.532231
Technical	SVM	0.546593	0.539810	0.570248
Linguistic based	LGBM dart 1000	0.606902	0.981043	0.110744
All features	LGBM dart 1000 est.	0.648391	0.991943	0.155372
Typing-behavior and Linguistic	Averaged outputs of best models	0.6208742	0.99575	0.09752

As it can be seen from Table 27, if one would like to build a classifier of the Twitter bad users, by typing behavior it is possible to detect around 10% of bad users with False Positive Rate around 1%, which is already satisfying for some of the tasks, as less possibility. As Typing-Behavior and Linguistic based models provide results, one of the model combination methods by averaging outputs of models was used, this led to decrease of False Negatives, Sensitivity is 0.995, however, the detection rate is lower and is less than 10%.

Moreover, if we unite all the features, it is possible to detect around 15% of a malicious users, only based on their username, while the False Positive Rate would remain less than 1%. Although 15% is not much, however, if such a score would be used as additional input with other features, it is possible to easily boost up detection preciseness.

With the email dataset, we see, that Typing Behavior and Linguistic-based models perform best, as well we can see that the Informational Similarity-based feature is not far away by its accuracy, while technical features left alone do not perform greatly (Table 28).

Table 28. Email dataset email address classification model performance

Model type	Model name	AUC	Sensitivity	Specificity
Typing-behavior	XGBoost	0.898516	0.974737	0.560871

Information-based	XGBoost	0.697749	0.976855	0.189427
Similarity (information)	XGBoost	0.856026	0.973766	0.484131
Technical	Extra tree	0.69	0.64	0.90
Linguistic based	XGBoost	0.856026	0.973766	0.484131
All features	XGBoost	0.953453	0.970851	0.701088
All without linguistic	XGBoost	0.949652	0.969879	0.686728
Averaged outputs	Typing-Behavior, Information, Similarity, and Linguistic	0.885916	0.991231	0.435379

However, the biggest accuracy is achieved when the model using all features is trained. A possible interpretation is that not only one set of characteristics describe the malicious or bad user, but it is also more combination of them, that is why it is better to exploit as much information can be gathered for classification.

Overall, it can be noted, that during classification, more accurately we can determine if the email is malicious comparing to the Twitter usernames, as the Twitter username has more limitation and malicious user might have to invest more time generating a new one, as it has to be unique in the whole database. In contrast, email does not have character limitation, if there is taken email, it is always easy for a malicious user to generate a new one.

Additionally, during the summary and conclusion formulation, a few more models were created, for example, one with all features except for linguistic, and as it can be seen from Table 28, such model metrics degraded a little – 2% fewer caught malicious users and 1% more False Negative results. Which is a trade-off that can be taken, to evade language bias. After performing the comparison of model outputs (Table 29), it was noted that all the best models, except for the best of all features and all features without linguistic features including the last one, provide independent results (all *p*-values are less than 0.05).

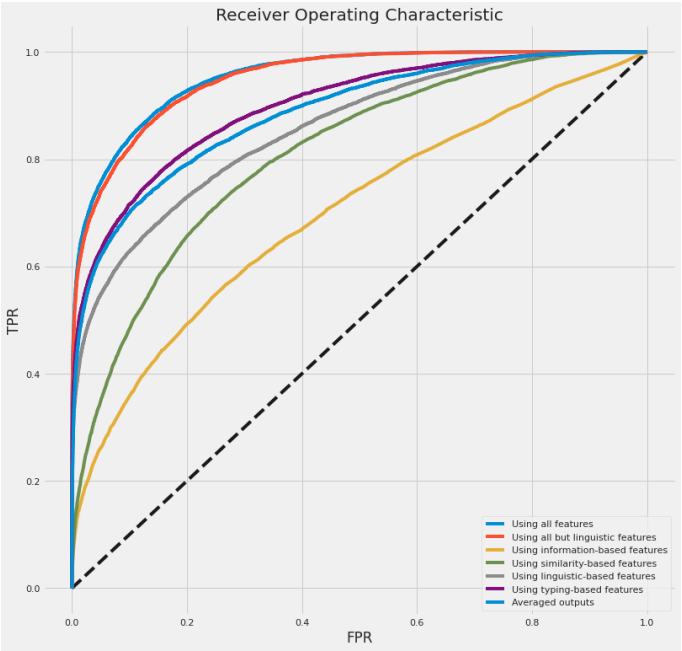

Fig. 40. Computed best email address classifier models

If we create the averaged model using model outputs of the Typing-Behavior, Information, Similarity, and Linguistic based features it is possible to achieve 0.9912 Sensitivity, which will lead to the False Negative Rate less than 1% and Specificity of 0.43. However, the AUC value will decrease comparing with All feature models, where with other than 0.5 cut-offs similar or better metrics can be achieved. Additionally, a similar RoC curve is for the classifier using only typing-behavior features.

Table 29. easyROC multiple comparison for best email address classifiers.

Marker1 (I)	**Marker2 (J)**	**AUC(I)**	**AUC(J)**	**|I - J|**	**SE(|I - J|)**	**z**	**p-value**	**p-value (adj.)**
typing_best	best_info	0.8985	0.6977	0.2008	0.0035	57.8229	0	0
typing_best	best_simi	0.8985	0.8086	0.0899	0.0029	31.1565	0	0
typing_best	linguistic_best	0.8985	0.856	0.0425	0.0027	15.5175	0	0
typing_best	all_best	0.8985	0.9535	0.0549	0.002	27.412	0	0
typing_best	all_without_ling	0.8985	0.9497	0.0511	0.002	25.2638	0	0
best_info	best_simi	0.6977	0.8086	0.1109	0.0038	29.3661	0	0
best_info	linguistic_best	0.6977	0.856	0.1583	0.0037	43.1853	0	0
best_info	all_best	0.6977	0.9535	0.2557	0.0032	81.0557	0	0
best_info	all_without_ling	0.6977	0.9497	0.2519	0.0032	79.5302	0	0
best_simi	linguistic_best	0.8086	0.856	0.0474	0.0031	15.2171	0	0
best_simi	all_best	0.8086	0.9535	0.1448	0.0025	58.0726	0	0
best_simi	all_without_ling	0.8086	0.9497	0.141	0.0025	56.1864	0	0
linguistic_best	all_best	0.856	0.9535	0.0974	0.0023	41.9494	0	0
linguistic_best	all_without_ling	0.856	0.9497	0.0936	0.0023	40.0157	0	0
all_best	all_without_ling	0.9535	0.9497	0.0038	0.0014	2.6922	0.0071	0.1065

For future research, it is possible to enrich a number of features, based on the technical aspect of emails and focus more on identifying fraudulent users in Social Networks. As well as the incorporation of such score into the regular detection models, as malicious users tend to reverse engineer used features, such non-dynamic feature could make harder for a malicious user to reverse engineer and successfully perform malicious attacks.

Additionally, as alone linguistic features were performing, adding geolocational features, such as login country, used languages in login country could also perform.

One more vector of improvement is using Neural Network model architecture, however, this will lower the interpretability of the algorithm. In contrast, after identification of the most important features, it is possible apart from the prediction to give some metrics about the username, to give more context for the person or algorithm interpreting classification results.

Summary and Conclusion

1. According to the literature review, there are a few core reasons why digitalization is happening: technological advancement, which simplifies entry into the digital world; policies, which promote forming digitalization strategy as part of the transformation into climate neutral world; economical, when companies are exploring new channels for products. However, with the digitalization process companies and entities face new challenges and threats, such as user identification and potential losses from malicious activity. The most dangerous threats which companies face are financial fraud (which projects direct financial losses) and attacks of "human bots" or regular bots for opinion manipulation or abuse of loyalty programs. Because of that, the most affected areas by malicious activity are e-commerce and social networks.
2. Analysis of current fraud prevention methodologies is performed. According to the reviewed literature, the best combination of methods for fraud prevention – feature-based fraud detection using supervised classification methods. This combination has the advantages of being easily scalable and requiring the least infrastructure or expenses. However, with the tendency of promoting privacy, gathering high-quality features for data scientists and machine learning engineers is becoming increasingly harder, resulting in decreased detection accuracy. That is why the exploration of additional data sources for feature engineering is constantly explored. One of the proposed new channels – evaluation of email or username formulation. As well, for all possible methods used in the fraud prevention industry – the proposed solution of email or username evaluation can be adopted, for additional accuracy.
3. During methodology, the following characteristics of the malicious user were defined: they try to be complex and diversify information, to make it harder to identify them; users are demographically biased; users try to be efficient and they are similar to other malicious users, but different from the good ones. Based on those characteristics there were proposed several methods and possible calculations of features. Later those additional inputs, which were calculated are used in different model training, such as Logistic Regression, Support Vector Machine, Decision Trees, and assembles of Decision Trees.
4. Two datasets were prepared for training: one consists of usernames of people discussing games on Twitter during American Football matches and a network of suspended and deleted users, which were seen during the #VoterFraud trend in November-December 2020. In total 13574 Twitter usernames were collected, of which 3051 are assumed to be fraudulent. Another dataset is the email dataset consisting of several data leaks and databases of fraudulent users provided by CleanTalk.org and reddit.com/r/datasets. The email dataset has in total 255352 emails, out of which more than 50 000 emails are fraudulent.
5. After feature engineering, different machine learning algorithms were applied for each dataset with different feature sets. Based on the training results, it is possible to state the following:

    - During the classification of the Twitter dataset, it is possible to detect around 16% of malicious users with False Negative Rate less than 6% using only Typing Behavior or Linguistic feature sets. Results with other separate feature group models were unstable.
    - If all features are used for the classification of Twitter users in one single model result improvements are observed, it becomes possible to detect around 15% of malicious users with False Negative Rate lower than 1%. Averaging outputs of Typing Behavior

and Linguistic features gives detection rate less than 10%, however, Specificity is more than 0.995.

- During the classification of email addresses, 4 out of 5 separate feature group models provide stable results, with detection rates ranging from 18% up to 56% and maintaining False Negative Rate in the range between 2% and 5%.
- For the email classification model using all features, the detection rate increases up to 70% with False Negative Rate less than 3%, however during model feature exploration bias towards certain languages was detected. Without the usage of linguistic features, the detection rate dropped to 68% and the False Negative Rate rose slightly above 3%. Usage of averaged outputs of models does not dramatically improve results and can be comparable with Typing-behavior model results.

Based on these results it is noticeable, that models verifying email addresses have a higher detection rate, as this can be explained through better dataset quality of email addresses. Another possible reason is that email addresses have fewer restrictions during creation and provide more email addresses, as each local part can be repeated with another provider. However, as the email dataset has a lot of Lithuanian emails, the model made bias towards detection of Lithuanian language as being a sign of not being a malicious user. After the removal of linguistic features, results are still better for email addresses.

After an email or username evaluation output given by the classification model – it can be used in other detection methods as a quality feature or alone if the False Negative Rate is acceptable.

List of references

1. FETTING, Constanze. THE EUROPEAN GREEN DEAL. . 2020.
2. List of top Fraud Detection Startups - Crunchbase Hub Profile. [online]. [Accessed 19 May 2021]. Available from: https://www.crunchbase.com/hub/fraud-detection-startups
3. S&P 500 Companies - S&P 500 Index Components by Market Cap. [online]. [Accessed 7 March 2021]. Available from: https://www.slickcharts.com/sp500
4. KUJALA, Valtteri and HALONEN, Raija. Business Growth Using Open Source e-Commerce and ERP in Small Business. In : ABRAHAM, Ajith, CHERUKURI, Aswani Kumar, MELIN, Patricia and GANDHI, Niketa (eds.), *Intelligent Systems Design and Applications*. Cham : Springer International Publishing, 2020. p. 147–158. ISBN 978-3-030-16657-1.
5. ZOHURI, Bahman. 21st Century Technology Renascence A Driven Impacting Factor For Future Energy, Economy, Ecommerce, Education, or Any Other E-Technologies. . 2020.
6. Global retail e-commerce market size 2014-2023 | Statista. [online]. [Accessed 9 May 2021]. Available from: https://www.statista.com/statistics/379046/worldwide-retail-e-commerce-sales/
7. MARKO MÄKI and TUIJA TOIVOLA. Global Market Entry for Finnish SME eCommerce Companies. Technology Innovation Management Review [online]. 2021. Vol. 11, no. 1. Available from: timreview.ca/article/1413
8. OBRENOVIC, Bojan, DU, Jianguo, GODINIC, Danijela, TSOY, Diana, KHAN, Muhammad Aamir Shafique and JAKHONGIROV, Ilimdorjon. Sustaining enterprise operations and productivity during the COVID-19 pandemic: "Enterprise effectiveness and sustainability model." Sustainability [online]. 2020. Vol. 12, no. 15. DOI 10.3390/su12155981. Available from: https://www.mdpi.com/2071-1050/12/15/5981Citation Key: su12155981tex.article-number: 5981
9. WADE, M and BJERKAN, H. Three proactive response strategies to COVID-19 business challenges. MIT Sloan management review. 2020.
10. RAPACCINI, Mario, SACCANI, Nicola, KOWALKOWSKI, Christian, PAIOLA, Marco and ADRODEGARI, Federico. Navigating disruptive crises through service-led growth: The impact of COVID-19 on Italian manufacturing firms. Industrial Marketing Management. 2020. Vol. 88, p. 225–237.
11. BESSARAB, Anastasiia, MITCHUK, Olha, BARANETSKA, Anna, KODATSKA, Natalia, KVASNYTSIA, Olha and MYKYTIV, Galyna. Social Networks as a Phenomenon of the Information Society. Journal of Optimization in Industrial Engineering. 2021. Vol. 14, no. 1, p. 35–42.
12. Digital 2019 Global Digital Overview (January 2019) v01. [online]. [Accessed 24 March 2021]. Available from: https://www.slideshare.net/DataReportal/digital-2019-global-digital-overview-january-2019-v01
13. 10 Facebook Statistics You Need to Know in 2021 [New Data]. [online]. [Accessed 24 March 2021]. Available from: https://www.oberlo.com/blog/facebook-statistics
14. Facebook MAU worldwide 2020 | Statista. [online]. [Accessed 9 May 2021]. Available from: https://www.statista.com/statistics/264810/number-of-monthly-active-facebook-users-worldwide/
15. CHOKHANI, Santosh, FORD, Warwick, SABETT, Randy, MERRILL, Charles R and WU, Stephen S. Internet X. 509 Public Key Infrastructure Certificate Policy and Certification Practices Framework. RFC. 2003. Vol. 3647, p. 1–94.
16. AMEEN, Nisreen, TARHINI, Ali, SHAH, Mahmood Hussain, MADICHIE, Nnamdi, PAUL, Justin and CHOUDRIE, Jyoti. Keeping customers' data secure: A cross-cultural study of cybersecurity compliance among the Gen-Mobile workforce. Computers in Human Behavior. 2021. Vol. 114, p. 106531.

17. CLAPPER, Danial and RICHMOND, William. SMALL BUSINESS COMPLIANCE WITH PCI DSS. Journal of Management Information & Decision Sciences. 2016. Vol. 19, no. 1.
18. HAUPERT, Vincent and GABERT, Stephan. Short Paper: How to Attack PSD2 Internet Banking. In : International Conference on Financial Cryptography and Data Security. Springer, 2019. p. 234–242.
19. GARFINKEL, Simson L. Email-based identification and authentication: An alternative to PKI? IEEE security & privacy. 2003. Vol. 1, no. 6, p. 20–26.
20. MON, Khin Lay and NAING, Thiri. SECURE E-MAIL SYSTEM USING HYBRID APPROACH OF TRIPLE-DES AND RSA ENCRYPTION ALGORITHMS. .
21. CHO, Daegon and KWON, K Hazel. The impacts of identity verification and disclosure of social cues on flaming in online user comments. Computers in Human Behavior. 2015. Vol. 51, p. 363–372.
22. BALDUZZI, Marco, PLATZER, Christian, HOLZ, Thorsten, KIRDA, Engin, BALZAROTTI, Davide and KRUEGEL, Christopher. Abusing social networks for automated user profiling. In : International Workshop on Recent Advances in Intrusion Detection. Springer, 2010. p. 422–441.
23. CHO, Daegon, KIM, Soodong and ACQUISTI, Alessandro. Empirical analysis of online anonymity and user behaviors: the impact of real name policy. In : 2012 45th Hawaii international conference on system sciences. IEEE, 2012. p. 3041–3050. ISBN 1-4577-1925-8.
24. AUFFRET, Patrice. SinFP, unification of active and passive operating system fingerprinting. Journal in computer virology. 2010. Vol. 6, no. 3, p. 197–205.
25. SHIVAKUMAR, Shailesh Kumar. Mobile Web Performance Optimization. In : Modern Web Performance Optimization. Springer, 2020. p. 79–103.
26. VASTEL, Antoine, LAPERDRIX, Pierre, RUDAMETKIN, Walter and ROUVOY, Romain. Fp-stalker: Tracking browser fingerprint evolutions. In : 2018 IEEE Symposium on Security and Privacy (SP). IEEE, 2018. p. 728–741. ISBN 1-5386-4353-7.
27. ALI, Mohammed Aamir, GROSS, Thomas and VAN MOORSEL, Aad. Investigation of 3-D secure's model for fraud detection. In : Proceedings of the 8th Workshop on Socio-Technical Aspects in Security and Trust. 2018. p. 1–11.
28. Dispute and fraud card monitoring programs. [online]. [Accessed 30 March 2021]. Available from: https://stripe.com/docs/disputes/monitoring-programs#vdmp
29. Sixth report on card fraud. [online]. [Accessed 9 May 2021]. Available from: https://www.ecb.europa.eu/pub/cardfraud/html/ecb.cardfraudreport202008~521edb602b.en.html#toc1
30. HAYASHI, Fumiko. Remote Card Payment Fraud: Trends and Measures Taken in Australia, France, and the United Kingdom. Payments System Research Briefing. 2020. P. 1–6.
31. Nilson Report | Promotion. [online]. [Accessed 9 May 2021]. Available from: https://nilsonreport.com/content_promo.php?id_promo=16
32. Cybersecurity. [online]. [Accessed 9 May 2021]. Available from: https://www.jpmorgan.com/commercial-banking/insights/cybersecurity
33. ALI, Mohammed Aamir and VAN MOORSEL, Aad. Designed to be broken: a reverse engineering study of the 3D secure 2.0 payment protocol. In : International Conference on Financial Cryptography and Data Security. Springer, 2019. p. 201–221.
34. DOUGLASS, Duncan B. An examination of the fraud liability shift in consumer card-based payment systems. Economic Perspectives. 2009. Vol. 33, no. 1.
35. MASTERCARD. Security Rules and Procedures—Merchant Edition. . 2019. P. 169.

36. Collins 2017 Word of the Year Shortlist - Collins Dictionary Language Blog. [online]. [Accessed 9 May 2021]. Available from: https://blog.collinsdictionary.com/language-lovers/collins-2017-word-of-the-year-shortlist/
37. ALEMANNO, Alberto. How to Counter Fake News? A Taxonomy of Anti-fake News Approaches. European Journal of Risk Regulation. 2018/03/22. 2018. Vol. 9, no. 1, p. 1–5. DOI 10.1017/err.2018.12. Cambridge Core
38. Fake news - Explore - Google Trends. [online]. [Accessed 9 May 2021]. Available from: https://trends.google.com/trends/explore?date=2014-04-09%202021-05-09&q=%2Fg%2F1210rwkh
39. CHAPMAN, James. The power of propaganda. Journal of Contemporary History. 2000. Vol. 35, no. 4, p. 679–688.
40. STUKAL, Denis, SANOVICH, Sergey, BONNEAU, Richard and TUCKER, Joshua A. Detecting bots on Russian political Twitter. Big data. 2017. Vol. 5, no. 4, p. 310–324.
41. HOWARD, Philip N., WOOLLEY, Samuel and CALO, Ryan. Algorithms, bots, and political communication in the US 2016 election: The challenge of automated political communication for election law and administration. Journal of Information Technology & Politics. 3 April 2018. Vol. 15, no. 2, p. 81–93. DOI 10.1080/19331681.2018.1448735.
42. RIZOIU, Marian-Andrei, GRAHAM, Timothy, ZHANG, Rui, ZHANG, Yifei, ACKLAND, Robert and XIE, Lexing. # debatenight: The role and influence of socialbots on twitter during the 1st 2016 us presidential debate. In : Proceedings of the International AAAI Conference on Web and Social Media. 2018. ISBN 2334-0770.
43. AHSAN, Mohammad and SHARMA, TP. Spams classification and their diffusibility prediction on Twitter through sentiment and topic models. International Journal of Computers and Applications. 2020. P. 1–11.
44. THOMPSON, Nicholas and VOGELSTEIN, Fred. Inside the two years that shook Facebook–and the world. Wired. URl: https://www. wired. com/story/inside-facebookmark-zuckerberg-2-years-of-hell. 2018.
45. ZAFARANI, Reza and LIU, Huan. 10 bits of surprise: Detecting malicious users with minimum information. In : Proceedings of the 24th ACM international on conference on information and knowledge management [online]. New York, NY, USA : Association for Computing Machinery, 2015. p. 423–431. CIKM '15. ISBN 978-1-4503-3794-6. Available from: https://doi.org/10.1145/2806416.2806535Citation Key: 10.1145/2806416.2806535publisher-place: Melbourne, Australia
46. ZHU, Yin, WANG, Xiao, ZHONG, Erheng, LIU, Nathan, LI, He and YANG, Qiang. Discovering spammers in social networks. In : Proceedings of the AAAI Conference on Artificial Intelligence. 2012. ISBN 2374-3468.
47. CAO, Qiang, SIRIVIANOS, Michael, YANG, Xiaowei and PREGUEIRO, Tiago. Aiding the detection of fake accounts in large scale social online services. In : 9th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 12). 2012. p. 197–210.
48. SALAZAR, A., SAFONT, G., SORIANO, A. and VERGARA, L. Automatic credit card fraud detection based on non-linear signal processing. In : 2012 IEEE international carnahan conference on security technology (ICCST). 2012. p. 207–212. Citation Key: 6393560
49. EFTHIMION, Phillip George, PAYNE, Scott and PROFERES, Nicholas. Supervised machine learning bot detection techniques to identify social twitter bots. SMU Data Science Review. 2018. Vol. 1, no. 2, p. 5.

50. STUKAL, Denis, SANOVICH, Sergey, TUCKER, Joshua A. and BONNEAU, Richard. For Whom the Bot Tolls: A Neural Networks Approach to Measuring Political Orientation of Twitter Bots in Russia. SAGE Open. 1 April 2019. Vol. 9, no. 2, p. 2158244019827715. DOI 10.1177/2158244019827715.
51. HODGE, Victoria and AUSTIN, Jim. A Survey of Outlier Detection Methodologies. Artificial Intelligence Review. October 2004. Vol. 22, no. 2, p. 85–126. DOI 10.1023/B:AIRE.0000045502.10941.a9.
52. rfc5322. *Internet Message Format* [online]. [Accessed 13 May 2021]. Available from: https://datatracker.ietf.org/doc/html/rfc5322
53. rfc6854. *Update to Internet Message Format to Allow Group Syntax in the "From:" and "Sender:" Header Fields* [online]. [Accessed 13 May 2021]. Available from: https://datatracker.ietf.org/doc/html/rfc6854
54. rfc5321. *Simple Mail Transfer Protocol* [online]. [Accessed 13 May 2021]. Available from: https://datatracker.ietf.org/doc/html/rfc5321
55. DELISI, Matt and VAUGHN, Michael G. Correlates of crime. The handbook of criminological theory. 2016. P. 18–36.
56. XIE, Yinglian, YU, Fang, ACHAN, Kannan, PANIGRAHY, Rina, HULTEN, Geoff and OSIPKOV, Ivan. Spamming botnets: signatures and characteristics. ACM SIGCOMM Computer Communication Review. 2008. Vol. 38, no. 4, p. 171–182.
57. LIN, Chengfeng, HE, Jianhua, ZHOU, Yi, YANG, Xiaokang, CHEN, Kai and SONG, Li. Analysis and identification of spamming behaviors in sina weibo microblog. In : proceedings of the 7th workshop on social network mining and analysis. 2013. p. 1–9.
58. KOLMOGOROV, Andrei Nikolaevich. Three approaches to the definition of the concept "quantity of information." Problemy peredachi informatsii. 1965. Vol. 1, no. 1, p. 3–11.
59. GRUNWALD, Peter and VITÁNYI, Paul. Shannon information and Kolmogorov complexity. arXiv preprint cs/0410002. 2004.
60. Typing with All Ten Fingers - df7cb.de. [online]. [Accessed 15 May 2021]. Available from: https://www.df7cb.de/projects/10finger/
61. NOYES, Jan. The QWERTY keyboard: A review. International Journal of Man-Machine Studies. 1983. Vol. 18, no. 3, p. 265–281.
62. PRESS, S James and WILSON, Sandra. Choosing between logistic regression and discriminant analysis. Journal of the American Statistical Association. 1978. Vol. 73, no. 364, p. 699–705.
63. Logistic Regression. [online]. [Accessed 25 May 2021]. Available from: http://faculty.cas.usf.edu/mbrannick/regression/Logistic.html
64. ABE, Shigeo. *Support vector machines for pattern classification*. Springer, 2005.
65. Kernel Trick in SVM. Kernel Trick can solve this issue using… | by Siddhartha Sharma | Analytics Vidhya | Medium. [online]. [Accessed 15 May 2021]. Available from: https://medium.com/analytics-vidhya/how-to-classify-non-linear-data-to-linear-data-bb2df1a6b781
66. Machine Learning Basics: Decision Tree From Scratch | by Devesh Poojari | Medium. [online]. [Accessed 16 May 2021]. Available from: https://deveshpoojari.medium.com/machine-learning-basics-descision-tree-from-scratch-part-i-4251bfa1b45c
67. Understanding Random Forest. How the Algorithm Works and Why it Is… | by Tony Yiu | Towards Data Science. [online]. [Accessed 16 May 2021]. Available from: https://towardsdatascience.com/understanding-random-forest-58381e0602d2
68. KEARNS, Michael. Thoughts on hypothesis boosting. *Unpublished manuscript*. 1988. Vol. 45, p. 105.

69. Top Machine Learning Algorithms, Frameworks, Tools and Products Used by Data Scientists | CustomerThink. [online]. [Accessed 16 May 2021]. Available from: https://customerthink.com/top-machine-learning-algorithms-frameworks-tools-and-products-used-by-data-scientists/
70. FAWCETT, Tom. An introduction to ROC analysis. *Pattern recognition letters*. 2006. Vol. 27, no. 8, p. 861–874.
71. THOMPSON, Ken. Programming techniques: Regular expression search algorithm. *Communications of the ACM*. 1968. Vol. 11, no. 6, p. 419–422.
72. GitHub - google/cld3. [online]. [Accessed 13 May 2021]. Available from: https://github.com/google/cld3
73. GitHub - dncR/easyROC: easyROC: an interactive web-tool for ROC analysis. [online]. [Accessed 23 May 2021]. Available from: https://github.com/dncR/easyROC

Appendices

1 Appendix. Twitter message format.


```
{
  "contributors": null,
  "truncated": false,
  "text": "vote for our fandom so that james and nadine would be beyond proud nine\n\n#PushAwardsJaDines",
  "is_quote_status": false,
  "in_reply_to_status_id": null,
  "id": 650477220952018944,
  "favorite_count": 0,
  "source": "<a href=\"https://about.twitter.com/products/tweetdeck\" rel=\"nofollow\">TweetDeck</a>",
  "retweeted": false,
  "coordinates": null,
  "timestamp_ms": "1443920829414",
  "entities": {
    "user_mentions": [],
    "symbols": [],
    "hashtags": [
      {
        "indices": [
          73,
          91
        ],
        "text": "PushAwardsJaDines"
      }
    ],
    "urls": []
  },
  "in_reply_to_screen_name": null,
  "id_str": "650477220952018944",
  "retweet_count": 0,
  "in_reply_to_user_id": null,
  "favorited": false,
  "user": {
    "follow_request_sent": null,
    "profile_use_background_image": true,
    "default_profile_image": false,
    "id": 2457742093,
    "verified": false,
    "profile_image_url_https":
"https://pbs.twimg.com/profile_images/571501757205721088/iBuTDWcN_normal.jpeg",
    "profile_sidebar_fill_color": "DDEEF6",
    "profile_text_color": "333333",
    "followers_count": 2636,
    "profile_sidebar_border_color": "FFFFFF",
    "id_str": "2457742093",
    "profile_background_color": "642D8B",
    "listed_count": 1,
```

```
    "profile_background_image_url_https": "https://abs.twimg.com/images/themes/theme10/bg.gif",
    "utc_offset": null,
    "statuses_count": 8406,
    "description": "Our love for James and Nadine is beyond infinity. ~ 1/millions of jds ~",
    "friends_count": 38,
    "location": "JaDine Spies Agents",
    "profile_link_color": "FFB8F7",
    "profile_image_url": "http://pbs.twimg.com/profile_images/571501757205721088/iBuTDWcN_normal.jpeg",
    "following": null,
    "geo_enabled": false,
    "profile_banner_url": "https://pbs.twimg.com/profile_banners/2457742093/1411186786",
    "profile_background_image_url": "http://abs.twimg.com/images/themes/theme10/bg.gif",
    "name": "JaDine Spies",
    "lang": "en",
    "profile_background_tile": true,
    "favourites_count": 1485,
    "screen_name": "JaDineSpies",
    "notifications": null,
    "url": null,
    "created_at": "Tue Apr 22 06:48:15 +0000 2014",
    "contributors_enabled": false,
    "time_zone": null,
    "protected": false,
    "default_profile": false,
    "is_translator": false
  },
  "geo": null,
  "in_reply_to_user_id_str": null,
  "lang": "en",
  "created_at": "Sun Oct 04 01:07:09 +0000 2015",
  "filter_level": "low",
  "in_reply_to_status_id_str": null,
  "place": null
}
```

2 Appendix. Typing features distribution and boxplots

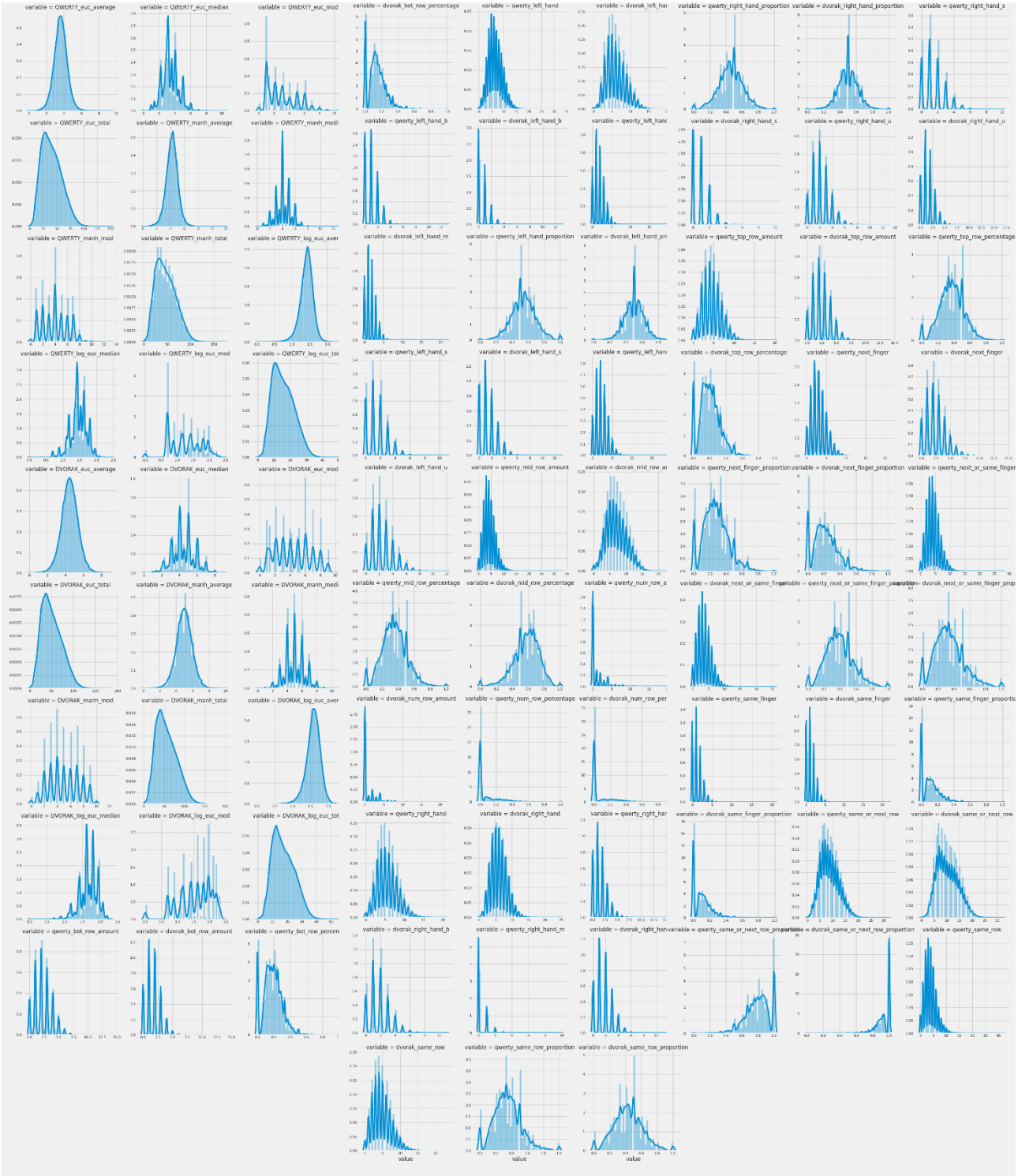

Fig. 41. Typing feature distribution plots for emails

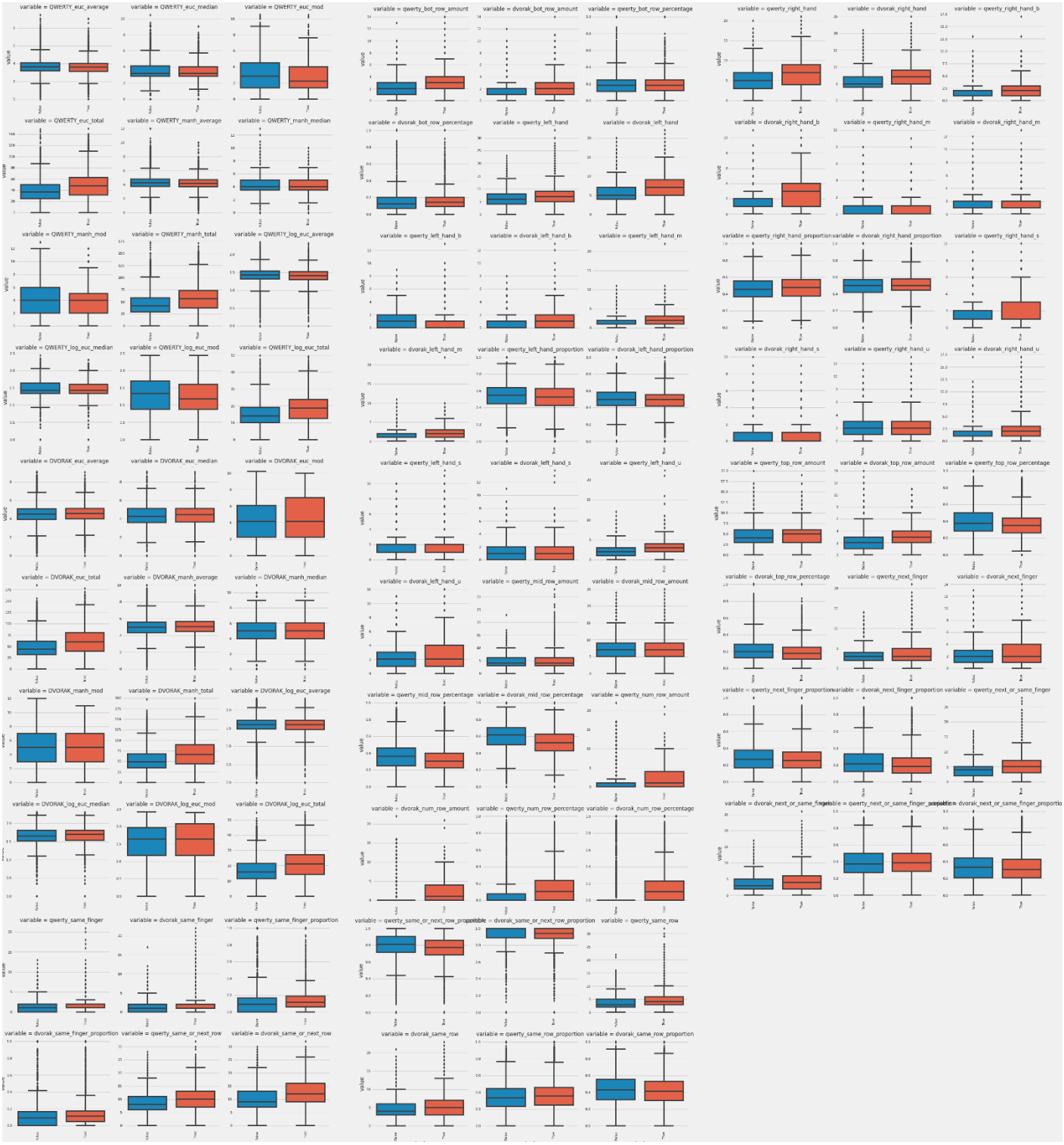

Fig. 42. Boxplots by variable bad for email datasets.

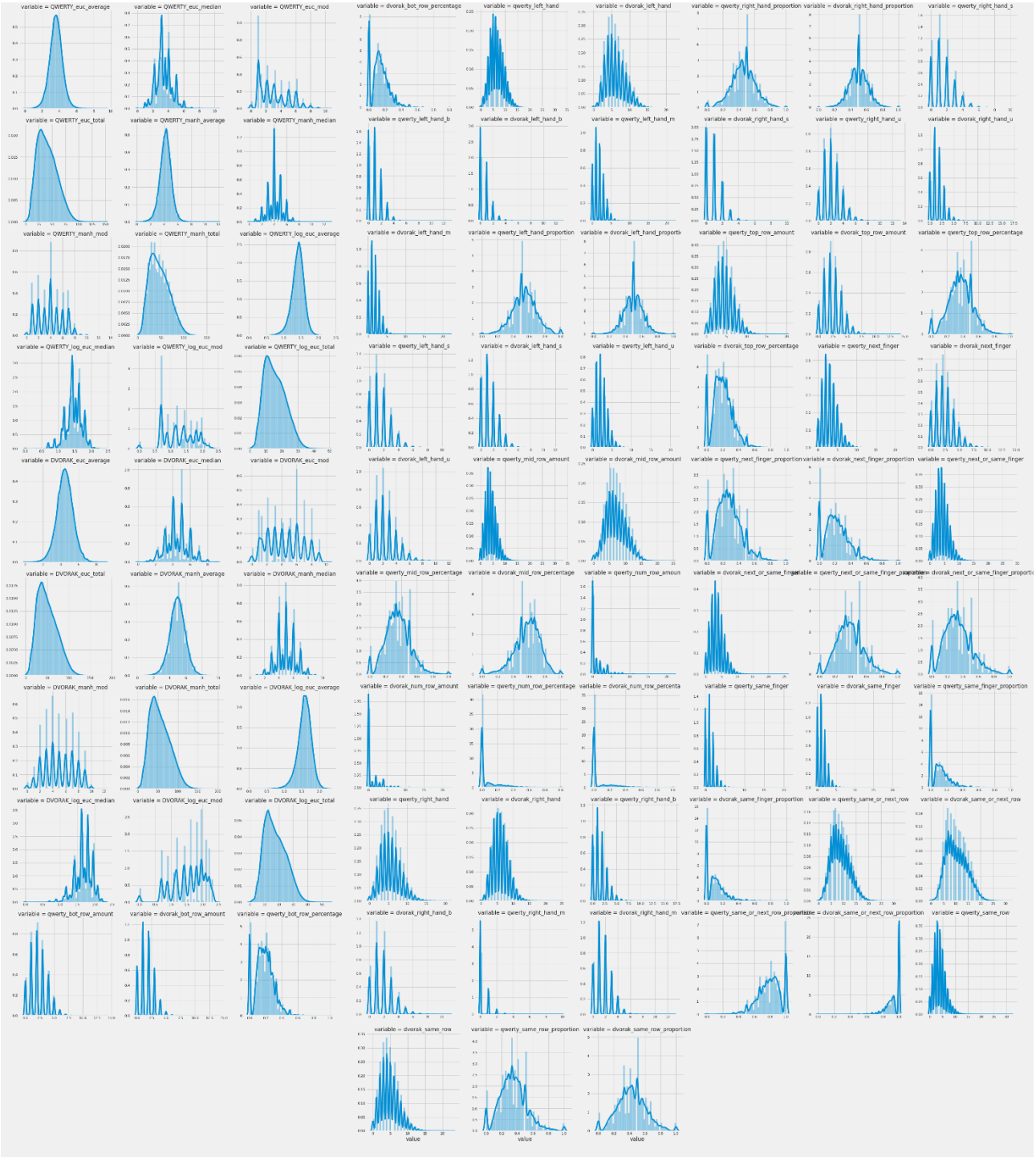

Fig. 43. Twitter Username typing features distribution

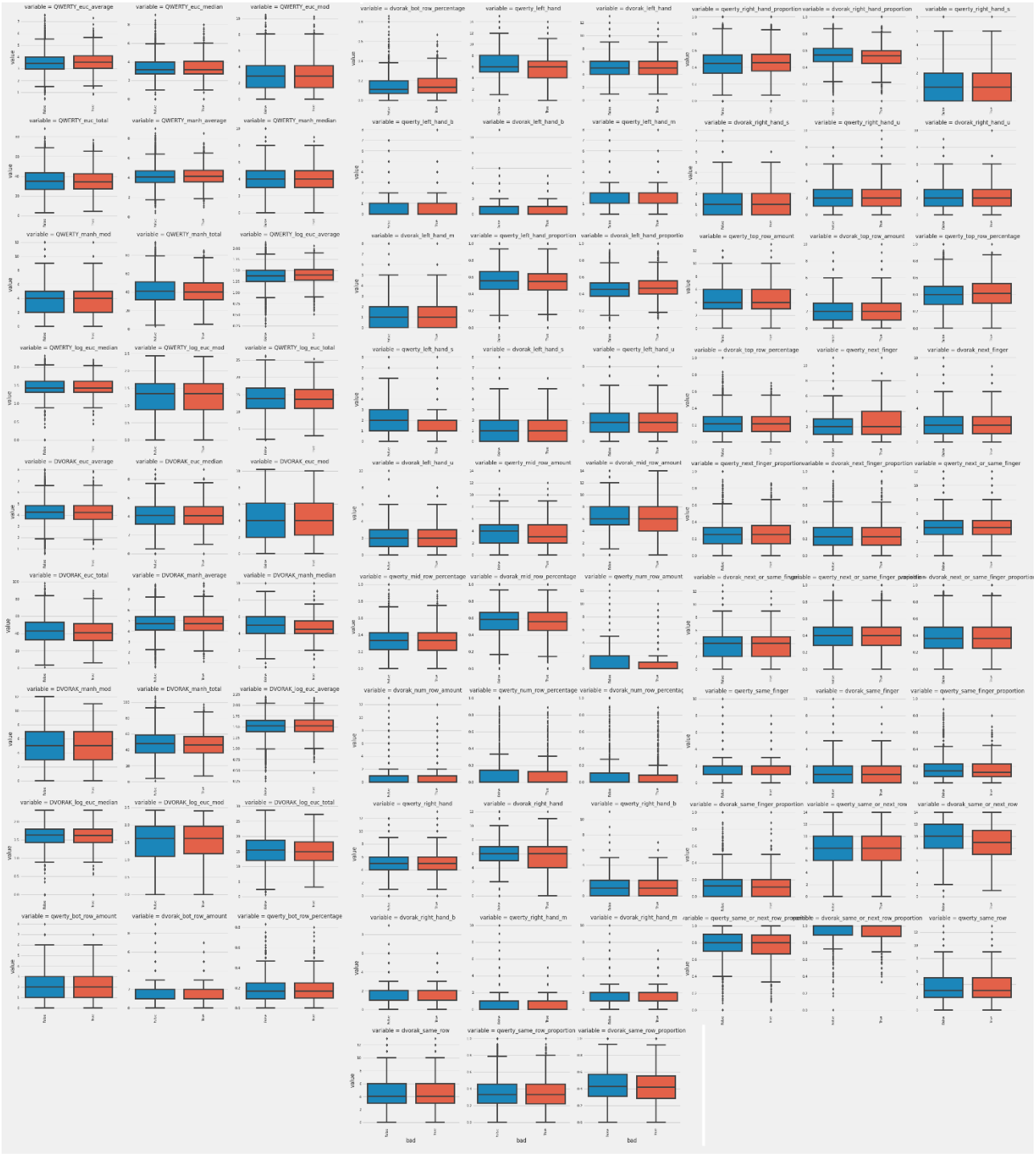

Fig. 44. Boxplots of typing features by value for Twitter usernames

3 Appendix. Feature importance graphs

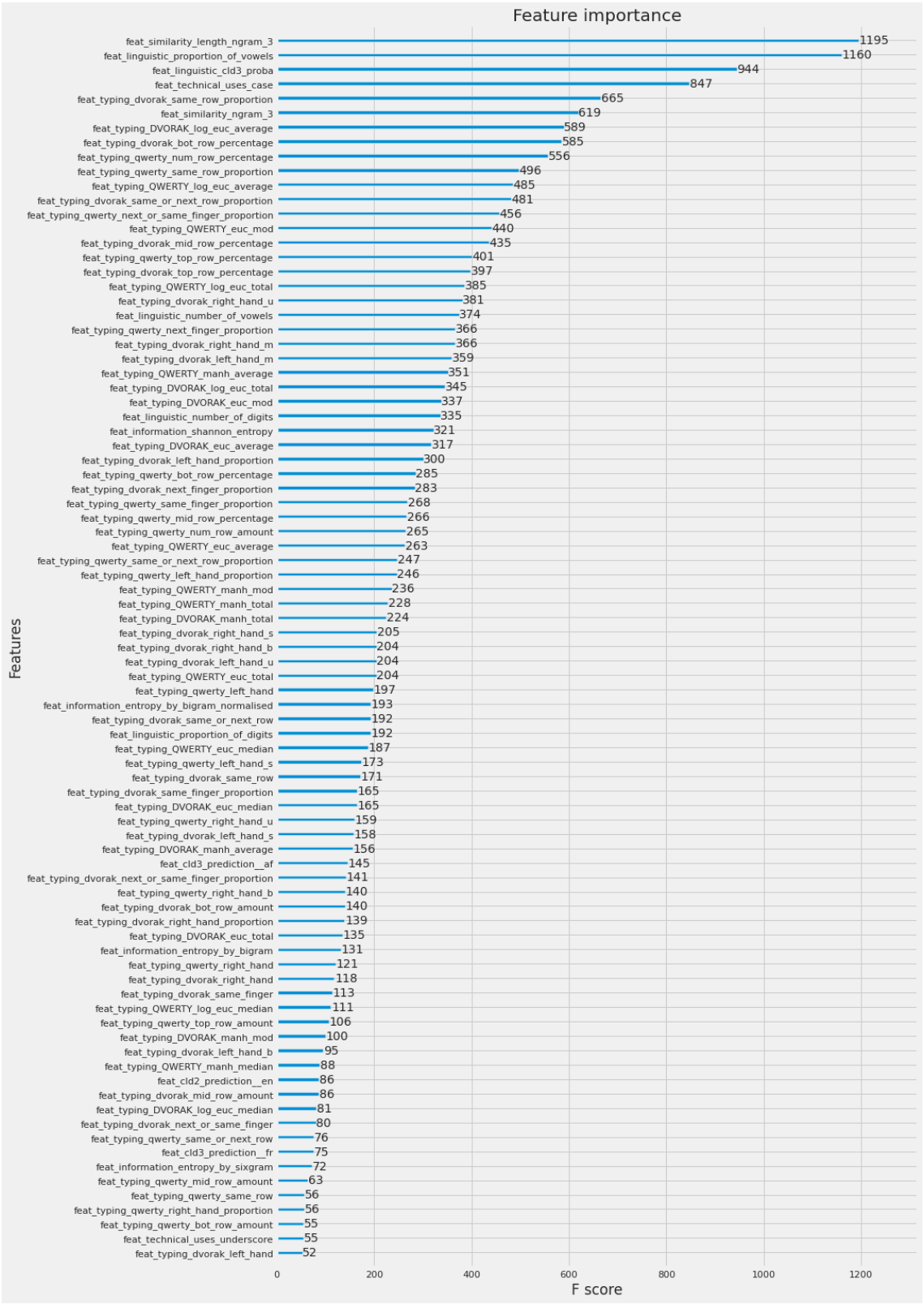

Fig. 45. Twitter Username. Model LGBM dart 1000 est. Feature importance graph, F score > 50

Fig. 46. Email address evaluation model XGB. Feature importance graph