August 16, 2019

MSC. THESIS

# PROFILING RECURSIVE RESOLVERS AT AUTHORITATIVE NAME SERVERS

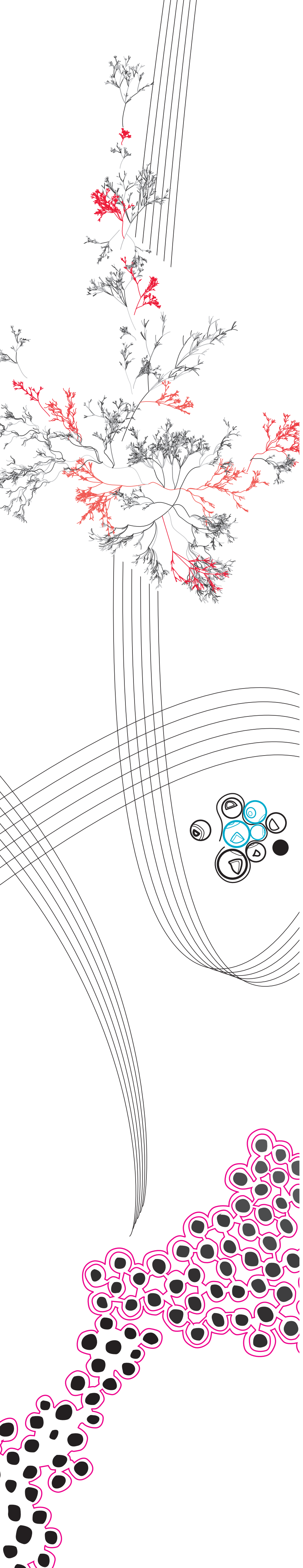
Metin A. Açıkalın - S1984853

**Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS)**

Exam committee:
dr. A. Sperotto (1st supervisor)
dr. M. Poel
ir. M.C. Muller (SIDN)

Department of Computer Science

UNIVERSITY OF TWENTE.



## Abstract

Domain Name System (DNS) translates a computer's fully qualified domain name into an IP address. Intermediary machines so-called recursive resolvers do this translation between a client and a DNS server. There are many recursive resolvers which connect to name servers every day. Each resolver show similarities and differences from one another. Knowing the origins of recursive resolvers can help to monitor significant operational changes in the DNS system and can be further used to prioritise some resolvers in case of DDoS attacks. There is too less study in the field which focuses on profiling them. In this thesis, standard behaviours of recursive resolvers and their behaviours in the wild are explained in detail. In addition, different classification methods applied to a data set consisted of 15 features to be able to classify recursive resolver origins in the case of .nl name servers. Random forest classifier had a 91% of overall accuracy predicting different resolver types on the dataset. According to the results of classification, more than 50% of unique resolvers contacting .nl name servers are originating from Internet Service Providers (ISPs). This is followed by open resolvers and cloud originating resolvers with around 10% and 7% respectively.

*Keywords: Domain Name System, Recursive Resolver, Classification, DNS, Classification in the Wild, Resolver Classification*

## Acknowledgements

Conducting this research and writing the thesis was a challenging process. Foremost, I would like to express my sincerest gratitude to my research supervisors Moritz C. Müller and Anna Sperotto for guiding me through this rough road; asking the most critical questions on shaping the research and being patient and supportive at all times.

Besides my supervisors, I would like to thank my small family; my mother Sibel Açıkalın, my father Şevki Açıkalın and my sister Elif Açıkalın for always being there for me not only in this thesis period but also every step that I take in my life. I wouldn't have succeeded any of these without their genuine support.

Furthermore, I sincerely thank from general to the specific to all members of SIDN and to all members of SIDN Labs for their hospitality and making me feel as if I am in my second home during my internship period in the company.

I would also like to thank my friends İrem Doğan, Sun Ok, Alejandro Dominguez, Dilek Başkaya, Ivan Lukman, Afet Çağay and Cristian van Herp for being by my side or being at the other end of the phone line to support me in every way they could. I don't know what would I do without you all!. Last but not least, I also want to thank Nicole Jansen for opening her home to me and becoming a good friend of mine in this short time.

# List of Abbreviations

| | |
|---|---|
| **AA** | Authoritative Answer |
| **APNIC** | Asia-Pacific Network Information Centre |
| **ASN** | Autonomous System Number |
| **ccTLD** | Country Code Top Level Domain |
| **CD** | Checking Disabled |
| **CNAME** | Canonical Name |
| **DNS** | Domain Name System |
| **DS** | Delegation Signer |
| **DSW** | data streaming warehouse |
| **HDFS** | Hadoop file-system |
| **IP** | Internet Protocol |
| **ISP** | Internet Service Provider |
| **MPP** | massively parallel processing |
| **MX** | Mail Exchance |
| **qmin** | query name minimisation |
| **RA** | Recursion Available |
| **RCODE** | Response Code |
| **RD** | Recursion Desired |
| **RIPE** | Réseaux IP Européens |
| **RR** | Resource Record |
| **SIDN** | Stichting Internet Domeinregistratie Nederland (in English: Foundation for Internet Domain Registration Netherlands) |
| **SOA** | Start of Authority |
| **SQL** | structured query language |
| **SRV** | Service locator |
| **TCP** | Transmission Control Protocol |
| **TTL** | Time To Live |
| **TXT** | Text Strings |
| **UDP** | User Datagram Protocol |
| **VPN** | Virtual Private Network |

# List of Figures

## List of Tables

*“Life isn’t about finding yourself.*
*Life is about creating yourself.”*

*George Bernard Shaw*

# Contents

# 1 Introduction

The Domain Name System (DNS) is a distributed, hierarchical naming system which translates a domain name into machine readible IP (Internet Protocol) addresses. DNS combines three major components [8, 9]:

- Domain namespace and resource records (RR), which are specifications for a tree-structured namespace and data associated with the names
- Name servers (NSes) which hold information about the domain tree's structure and provides responses to queries according to their ledgers.
- Resolvers which extracts information from name servers on behalf of their users

A simplified look of the name resolution process can be seen in Figure 1. If a client's operating system or web browser wants to use the DNS service, a query is sent via a stub resolver to a recursive resolver. In the example of Figure 1 the client wants to connect to example.com. Therefore, the stub resolver of the client creates a request to its recursive resolver, which can be seen as the first step in the figure. Then the resolver iteratively travels the levels of the DNS hierarchy starting from the root until it resolves the Internet Protocol (IP) address of example.com. These iterative searches can be seen in steps from 2 to 6. In the end, a DNS server returns the proper records (step 7), which then forwarded all the way back to the client, as stated in step 8. Finally, the client obtains the IP address of the server it wants to connect and uses this IP address to connect to the preferred domain, which can be seen in steps 9 and 10.

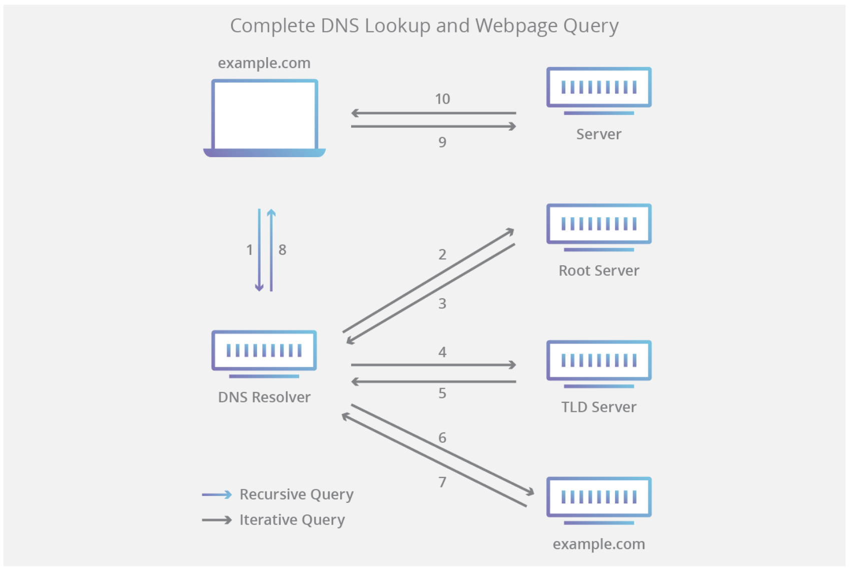


Figure 1: Visual representation of how DNS works [1]

## 1.1 Problem Statement

Authoritative NSes are designed to reply to the resolver queries. However, management and operation of Authoritative NSes could be improved if the type of resolver contacting the name server could be classified. According to DNS standards [8], all clients which reach to NSes should be DNS resolvers acting on behalf of their users. Despite, this is not always the case in a real-world environment. In a similar resolver classification case run on .nz name servers [10], the operators found that there are some records of known IP addresses that were not recursive resolvers acting on behalf of their users, but they were monitoring tools or up-time probes. However, they did not disclose absolute numbers. More details on this study will be discussed later in Section 2.5

Some of the impacts of detecting the resolvers at a name server is as follows:

- Being able to know which resolvers are directly relevant for end-users would allow operators to understand how they should set up their server infrastructure to serve those resolvers best. To illustrate, operators of NSes can build their servers physically closer to important resolvers such as resolvers of local Internet Service Providers (ISPs).
- In case of major operational changes, adoption of resolvers to these changes can be monitored better. To illustrate since 11 October 2018, a new key is used to sign the root zone which was created on 27 October 2016. This was a huge operational change, and it was known that many resolvers didn't have the newest key configured because of the DNSSEC validation errors. The operators of the root zone didn't know if there was a need to worry about these resolvers after this key rollover because the origins of these resolvers were unknown [11]. If the origins of recursive resolvers are known, it is easier to monitor the adaptations of huge operational changes like this by sector.
- Similarly, the administrators of name servers would be able to understand which resolvers should be prioritised in case these name servers are under a DDoS attack and have only limited resources to answer queries[1].
- The administrators of name servers would be able to raise an alert to the operators of resolvers if some of the important resolvers suddenly stop resolving or behaving oddly. This is important for the administrators of these recursive resolvers.

## 1.2 Research Objective & Questions

DNS has a complex structure. A recent study done by Müller et al. [2] shows this complex environment with Figure 2 for the .nl NSes case. It can be seen from the figure that a client can use two or more different upstream recursive resolvers for the same query or there can be a forwarding resolver, which is indicated as middleboxes in the figure, between client's resolver and authoritative NS. Besides, not all queries reach to an NS if the same recursive resolver has already resolved the domain within a specific time interval and the desired IP address of requested domain name is in its cache. On the other hand, it can also be observed from the figure that resolvers can select between multiple NSes. To be able to provide a better service to these recursive

[1]Prioritisation here does NOT mean -not serving- to some types resolvers, but deciding the distribution of remaining resources over resolver types.

resolvers, profiling them is one of the useful methods considering the aforementioned complex environment of DNS.

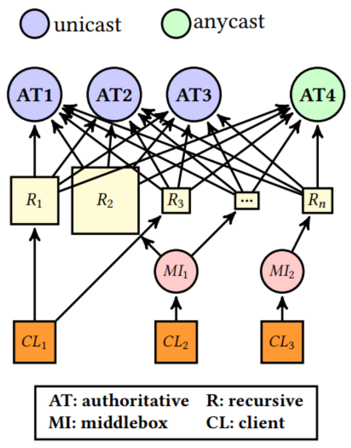


Figure 2: TLD Setup, Recursives, Middleboxes and Clients. [2]

While conducting this research, my main point is going to be the classification of recursive resolvers at Authoritative NSes. In this research, quantitative research techniques are going to be used.

The research questions of the thesis are as follows:

- Research Question 1: What is the expected behaviour of recursive resolvers? This part is the main focus of the Literature Review in Section 2 to be able to understand the standards of the recursive resolvers. Finding this out will also make it easier to select features for the classification of these resolvers.
- Research Question 2: How to classify the recursive resolvers at an authoritative name server? All the necessary steps for the machine learning case and different machine learning algorithms to classify recursive resolvers will be covered. I expect to see different classes such as ISP resolvers, open resolvers, cloud resolvers, and so on.
- Research Question 3: What are the main recursive resolvers of .nl NSes? By being able to identify this, the study will gain a real-world example on the proposed model. Which feature types can be useful to distinguish different types of resolvers are also mentioned in the thesis paper.

In my thesis paper, in Section 2, I will provide relevant studies to be able to use the most recent techniques for profiling the recursive resolvers for this research. In the remaining part of the paper, in Section 3 I will go further into the methodology and define my working environments. Furthermore, I will also explain followed methods to identify the feature set and profiling recursive resolvers from the knowledge of standard resolver behaviour and their behaviours in the wild. Then, in Section 4, I will provide details on the results of the research. I will finish the paper in Section 5, where I will provide a discussion of my results and provide conclusions.

# 2 Literature Review

In this section, I will answer the first research question on “what is the expected behaviour of recursive resolvers” to be able to use the information shared in this section to create a feature set which then will be used in classification. The structure of this section will be reviewing related papers on the topics of recursive resolver behaviour and machine learning techniques for classification purposes. After achieving this goal, how to classify such behaviours on a set of features can be discussed on a concrete base. Furthermore, as machine learning applications will be used in this research, more information on classification algorithms such as their methods and advantages will also be explained.

## 2.1 Standard Resolver Behaviour

In Request For Comments (RFC) 1034 published in 1987, the main points of how a DNS should set-up and how it should systematically work are explained [8]. According to it, recursive design in DNS is highly essential for several reasons. One of the reasons mentioned in RFC which is highly relevant for this research is that recursive design is necessary for a simple requester which can not do anything else other than receiving a direct answer to the query which is often called a “stub resolver”. Furthermore, it is also crucial for a network where one wants to concentrate the cache rather than having a separate cache for each client. By this way, multiple requests from distinct clients of the same network can get replies faster as the answer to the query will be already in the cache. Therefore, time and space resources will be used more efficiently.

To be able to use recursion between DNS server and client, an agreement is proposed to be made between them. According to the procedure, for this agreement, there are two-bit fields, namely Recursion Desired (RD) and Recursion Available (RA) flags. If a resolver wants to use recursion, the RD flag is set in the query. The agreement is completed if also the Authoritative NS sets RA flag in response to that query. The recursive mode occurs when a query with RD set arrives at an NS which is willing to provide recursive service; the client can verify that recursive mode was used by checking that both RA and RD flags are set in the reply [8]. This can be observed in Figure 1 marked with 1 and 8. The communication between client and recursive resolver is recursive with the help of these flags. Therefore if this flag is set in a query seen in a ccTLD NS, this means that the resolver contacting the NS is either a stub resolver or a resolver which is not conforming to the standards.

Another point that is important for this research which defined in the RFC 1034 is the fact that not all resolver requests that are sent from clients to recursive resolvers are seen on an Authoritative NS. This is because of the Time To Live (TTL) value that is included in the DNS query response. In RFC 1034, Time To Live (TTL) is defined as a field that is “a 32-bit integer in units of seconds, an is primarily used by resolvers when they cache RRs. The TTL describes how long a RR can be cached before it should be discarded” [8]. For .nl authoritative NS, this time is set to 3600 seconds, which corresponds to an hour. For example, if a correctly configured recursive resolver contacts to any .nl NS for name resolution and receives another request on the same domain name within an hour, it is expected from that resolver not to contact NS of .nl for name resolution. However, the same information stored in different levels of DNS hierarchy can have different TTL values. Then, a resolver should respect the TTL value of the child NS.

## 2.2 Recursive Resolver Algorithm

### 2.2.1 Resolver Behaviour According to the RFCs

To be able to classify the recursive resolvers according to the behaviours, knowing the algorithm of them, which describes how they work, is important. By this way, why and how the features are selected can be understood better. In RFC 1034 [8], the algorithm of resolvers is described as follows:

1. Look for the queried record in the local cache, if found, return the answer.
2. Find the best servers to ask. This is done by trying to find an authoritative answer providing servers for the requested query.
3. Send the query to the servers until one returns a response.
4. Analysis part of the received response:
    (a) If the response is the answer of the query or if it contains a name error, cache the response and return it to the client.
    (b) If the response is including better delegation to other name servers, cache the delegation information and return to step 2.
    (c) If the response is showing a CNAME, cache the CNAME, change the query to what canonical name is pointing to and return to step 1.
    (d) If the response is showing server failure message or other unknown content, delete the server from the SLIST[2] and return to step 3.

Internet Assigned Numbers Authority (IANA) has a hint file on their website which points thirteen well-known root name servers' IP addresses for the operators of this recursive resolvers as a starting point for configuration. That is how recursive resolvers know about root (.) name server IP addresses and then iteratively learn about TLD addresses [12].

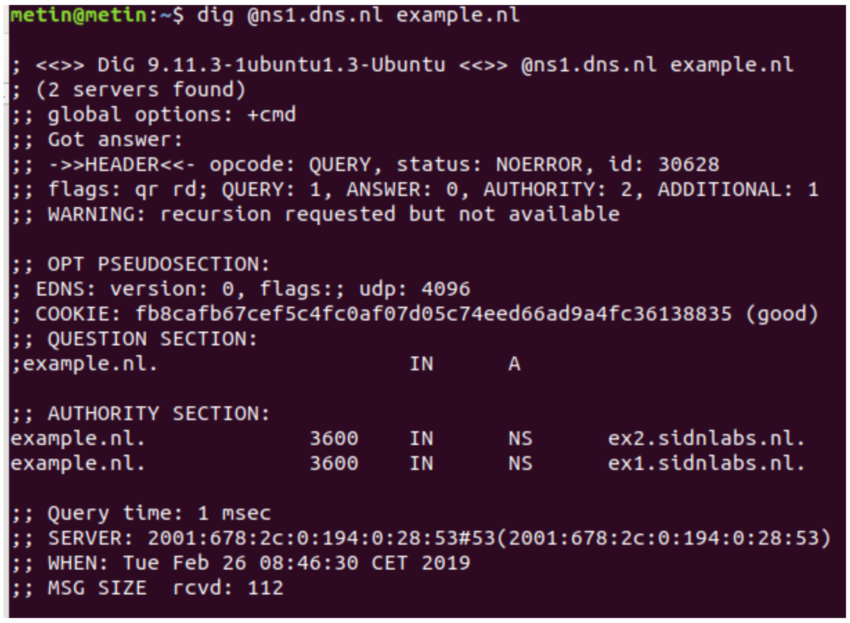
### 2.2.2 Example Lookup Scenarios for Standard Behaviour

It is important to understand the algorithmic relation between a recursive resolver and authoritative NS. In this section, different lookup scenarios from the perspective of .nl ccTLD authoritative NS and recursive resolver will be shared. These example queries will provide a concrete base on how an NS answers a resolver's queries. It is also important to indicate that all these resolutions are directly asked to .nl NSes and

[2]The structure which keeps track of the resolver's current best guess about which name servers hold the desired information; it is updated when arriving information changes the guess [8].

all the variables which can vary NS to NS, such as time to live value of an answer, are in the example of .nl NSes.

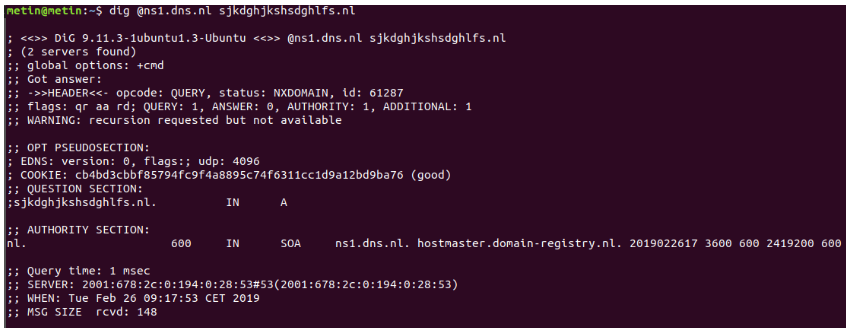
**Lookup of an existing domain name:** Assume that a resolver queries “A record” of example.nl and name server of example.nl is also in .nl NS. In Figure 3 query and answer for this situation can be seen. What happens in the .nl NS side is that an answer is created that points the authoritative name server of example.nl which is *ex1.sidnlabs.nl* & *ex2.sidnlabs.nl* in this situation and TTL will be set to 3600 seconds. Whenever the recursive resolver receives the answer, it stores this information in its cache for an hour unless there is another rule set in resolver that overrules the TTL section of the answer sent by authoritative NS. Finally, the resolver will get in touch with one of the authoritative NS of example.nl and will forward the IP address of example.nl to the client.



```
metin@metin:~$ dig @ns1.dns.nl example.nl

; <<>> DiG 9.11.3-1ubuntu1.3-Ubuntu <<>> @ns1.dns.nl example.nl
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 30628
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 2, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: fb8cafb67cef5c4fc0af07d05c74eed66ad9a4fc36138835 (good)
;; QUESTION SECTION:
;example.nl.                    IN      A

;; AUTHORITY SECTION:
example.nl.             3600    IN      NS      ex2.sidnlabs.nl.
example.nl.             3600    IN      NS      ex1.sidnlabs.nl.

;; Query time: 1 msec
;; SERVER: 2001:678:2c:0:194:0:28:53#53(2001:678:2c:0:194:0:28:53)
;; WHEN: Tue Feb 26 08:46:30 CET 2019
;; MSG SIZE  rcvd: 112
```

Figure 3: Answer of .nl ccTLD name server for example.nl

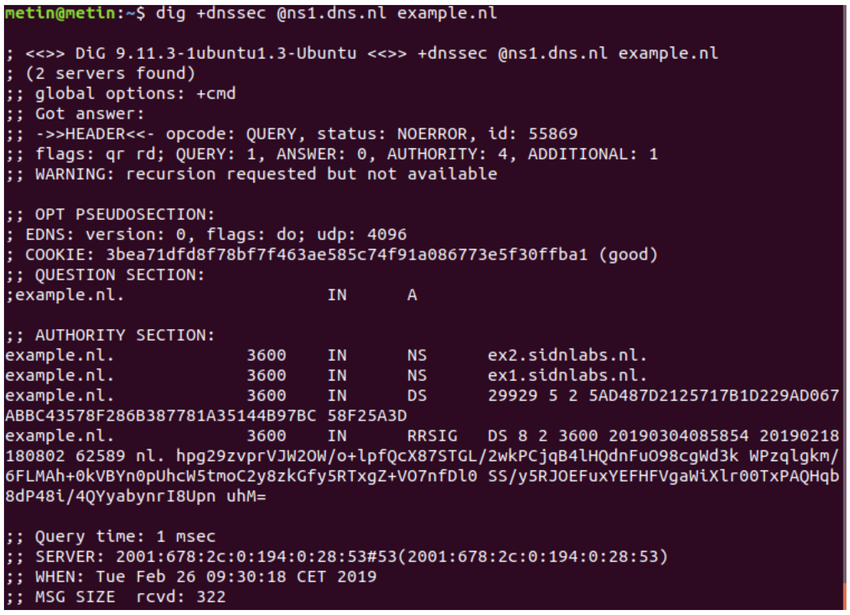
**Lookup of a non-existing domain name:** This time suppose that a resolver queries “A record” of a random website that does not exist, for instance, sjkdghjkshsdghlfs.nl. This means that the name server of sjkdghjkshsdghlfs.nl is not in .nl NS zone. Then at the .nl NS side, a Non-Existent Domain (NXDomain) answer will be created as can be seen in the status part of Figure 4. This time TTL will be set to 600 seconds as NXDomain the TTL standards of .nl NS is 600 seconds. On the resolver side, the answer will be cached for 600 seconds as NXDomain answer unless there is another rule set in resolver that overrules TTL section of the answer sent by authoritative NS. Finally, the NXDomain answer will be forwarded to the client.



```
metin@metin:~$ dig @ns1.dns.nl sjkdghjkshsdghlfs.nl

; <<>> DiG 9.11.3-1ubuntu1.3-Ubuntu <<>> @ns1.dns.nl sjkdghjkshsdghlfs.nl
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 61287
;; flags: qr aa rd; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: cb4bd3cbbf85794fc9f4a8895c74f6311cc1d9a12bd9ba76 (good)
;; QUESTION SECTION:
;sjkdghjkshsdghlfs.nl.          IN      A

;; AUTHORITY SECTION:
nl.                     600     IN      SOA     ns1.dns.nl. hostmaster.domain-registry.nl. 2019022617 3600 600 2419200 600

;; Query time: 1 msec
;; SERVER: 2001:678:2c:0:194:0:28:53#53(2001:678:2c:0:194:0:28:53)
;; WHEN: Tue Feb 26 09:17:53 CET 2019
;; MSG SIZE  rcvd: 148
```

Figure 4: Answer of .nl ccTLD name server for sjkdghjkshsdghlfs.nl

**Lookup of a domain name secured with DNSSEC:** Another scenario can be that a resolver queries "A record" of example.nl. Also, assume that example.nl is signed with DNS Security Extensions (DNSSEC), and a resolver does DNSSEC validation. On the resolver side, with DNSSEC, the resolver will not only resolve the "Domain Name - IP address" pair but also validate the cryptographic signatures gathered from authoritative NS to ensure that the DNS information was not modified in transit. To be able to do it, resolver also gets RRSIG information which can be seen in Figure 5 and iteratively checks so-called "Chain of Trust" starting from the name server of example.nl and going up in the hierarchy to ccTLD of example.nl which is .nl and finally it will end up at the beginning of the trust at a root (.) NS. This process can be followed from the Figure 6. In the figure representation, digital signature (DS record) attached to the signer's public key, which is #1 (DNSKEY record) confirms the authenticity of the signer's signatures. Moreover, a digital signature attached to the public key (#2) of the signer of that public key (#1) confirms the authenticity of that public key (#2). Hence, a 'chain of trust' is created within the DNS infrastructure, anchored in the root zone [3]. The resolvers which can follow this process to validate the integrity of the resolution are called "DNSSEC-validating DNS resolvers". They resolve DNS domains that are DNSSEC-signed and validated correctly (AD flag) and reject DNS domain with broken DNSSEC are not validated (SERVFAIL). They also allow non-DNSSEC-signed domains to resolve [13]. So, on the ccTLD NS side, we do not expect the same resolver to connect to the server again in an hour for DNSSEC validation for the same domain name. This is again because the TTL values of DNSKEY, DS and RRSIG records, which are the typical RR query types for DNSSEC validating resolvers, are all set to 3600 seconds which can be seen in Figure 5



```
metin@metin:~$ dig +dnssec @ns1.dns.nl example.nl

; <<>> DiG 9.11.3-1ubuntu1.3-Ubuntu <<>> +dnssec @ns1.dns.nl example.nl
; (2 servers found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 55869
;; flags: qr rd; QUERY: 1, ANSWER: 0, AUTHORITY: 4, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
; COOKIE: 3bea71dfd8f78bf7f463ae585c74f91a086773e5f30ffba1 (good)
;; QUESTION SECTION:
;example.nl.                    IN      A

;; AUTHORITY SECTION:
example.nl.             3600    IN      NS      ex2.sidnlabs.nl.
example.nl.             3600    IN      NS      ex1.sidnlabs.nl.
example.nl.             3600    IN      DS      29929 5 2 5AD487D2125717B1D229AD067
ABBC43578F286B387781A35144B97BC 58F25A3D
example.nl.             3600    IN      RRSIG   DS 8 2 3600 20190304085854 20190218
180802 62589 nl. hpg29zvprVJW2OW/o+lpfQcX87STGL/2wkPCjqB4lHQdnFuO98cgWd3k WPzqlgkm/
6FLMAh+0kVBYn0pUhcW5tmoC2y8zkGfy5RTxgZ+VO7nfDl0 SS/y5RJOEFuxYEFHFVgaWiXlr00TxPAQHqb
8dP48i/4QYyabynrI8Upn uhM=

;; Query time: 1 msec
;; SERVER: 2001:678:2c:0:194:0:28:53#53(2001:678:2c:0:194:0:28:53)
;; WHEN: Tue Feb 26 09:30:18 CET 2019
;; MSG SIZE  rcvd: 322
```

Figure 5: Answer of .nl ccTLD name server including DNSSEC records for example.nl

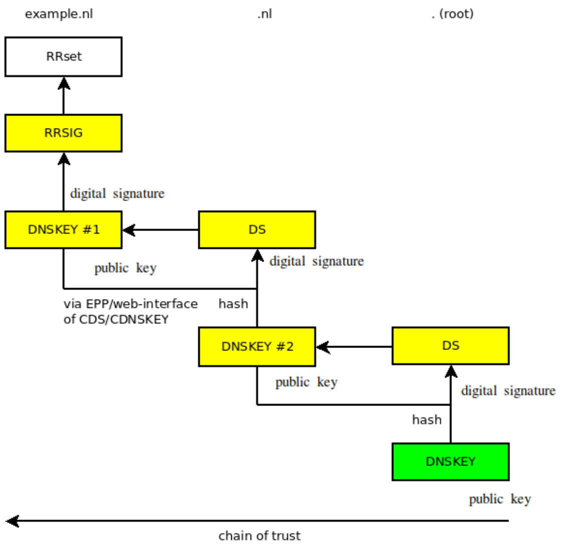


Figure 6: Chain of trust in example.nl example [3]

## 2.3 Resolver Behaviour in the Wild

In the past 32 years from the first published RFC on DNS in 1987, some dynamics have changed including domain name resolving process by recursive resolvers and security of DNS. To illustrate, when DNS was first proposed, security was not even a consideration at the time. The main purpose of RFC 1034 was to get things working. A couple of years later, researchers started mentioning the security of DNS and publishing these by extensions on the security of DNS like in RFCs 2535, 3007, 3008 [14, 15, 16].

Resolving process of a domain has also changed during the years. The process which was described in Section 2 with RD and RA fields are not much in use anymore.

How a recursive resolver resolves a fully-qualified domain is described in a research published in 2015 by Kührer et al. [17]. A threat model which affects clients that use and blindly trust DNS resolvers is explained in the paper. They indicated that they found millions of resolvers which deliberately manipulate DNS resolutions and added that these resolvers may or may not return correct recursive DNS resolutions. Then, they defined the "correct" recursive resolvers as resolvers, which strictly follow the hierarchy for DNS lookup. This means starting at the root (.) servers then following the Top Level Domain (TLD) (*e.g.,* .nl) and then iteratively querying the Authoritative NSes of a domain name to resolve fully-qualified domain (*e.g.,* www.example.nl). Therefore it can be concluded that the only responsible units in the Domain Name System to recursively follow the hierarchy, are resolvers. Authoritative NSes do not set RA flag in DNS responses to help resolvers to find IP addresses of the domain names which they are not authoritative to.

### 2.3.1 Name Server Choice: How It is Done?

The previous work on a ccTLD NS conducted by Müller et al. explain how a recursive resolver makes choices for which authoritative name server to connect if there are more than one authoritative name server IP addresses [2]. According to the paper most recursive resolvers (75 to 96%) query all authoritatives and they choose their "preferred" one according to the Round Trip Time (RTT)[3] values of the queries. Even though the study conducted to show how choices of recursive resolvers are made in the wild; for this research, it is important to conclude that we might not see all of the ccTLD name server traffic if we are looking to one of the name server's traffic.

### 2.3.2 Forwarding Resolvers & Resolver Pools

On the other hand, not all resolvers are recursive resolvers. In a research conducted on the rise of a malicious resolution authority, another type of resolver, the forwarding resolvers, are mentioned [18]. In the paper, Dogon et al. specified that they stored the IP addresses of the open recursive resolvers that they asked to resolve a query and the IP addresses that contacted to their authoritative NS for that query. Then they tried to match the IP addresses they sent their queries to, with the addresses their name servers -which was authoritative for that domain- get queried for that address. They indicated that 96.4% of the queries are resolved by another IP address than the asked one, which brings us to the conclusion of forwarding resolver existence.

In addition to this, in a paper, published in 2018 [19], they discovered pools of recursive resolvers acting all together behind one interface on different IP addresses. This also can be the reason for different IP address pairs for queried resolver and the resolver who reaches to the authoritative NS for the same query. They found that most pools are small with 38.7K (63%) of pools contain two resolvers. They have seen that 21.5K (35%) pools with two resolvers contain one IPv4 and one IPv6 address. The largest pool they discovered consisted of 317 IP addresses contained within 5 IPv4

[3]The time needed for a signal to reach from an origin to a specific destination and coming back to origin.

/24 CIDR (Classless Inter-Domain Routing) blocks and 8 IPv6 /64 CIDR blocks. All blocks belong to ASN 15169, Google Inc. In all, 85% of the pools consist of less than ten resolvers.

To conclude, it is possible for an IP address which connected to an NS for a resolution is not the original IP address of the machine which the query is sent to. Therefore, this also means that one resolver might spread its queries across multiple IP addresses. This can result in analysing a single behaviour distributed into different IP addresses, which can reduce the accuracy of the classification algorithm as the classifier will consider each IP address as a unique resolver.

### 2.3.3 QNAME Minimisation

Another property that has changed in the wild in recent years is that some of the recursive resolvers started doing query name minimisation (qmin) as standardised in RFC 7816 [20]. How this process is working is explained in the Table 1 from a recent study [7]. In the table, both standard behaviour, which sends all the query from the user, and qname minimised queries, which just sends the necessary information for each level of the hierarchy, can be seen. By applying this, admins of root NSes or TLD NSes can not know what the full query requested by the end-user is. This brings privacy to end-users.

| Standard DNS Resolution | | qmin Reference (RFC 7816) | |
|---|---|---|---|
| a.b.example.com | A $\rightarrow .$ | com. | NS $\rightarrow .$ |
| com. | NS $\leftarrow .$ | com. | NS $\leftarrow .$ |
| a.b.example.com | A $\rightarrow com.$ | example.com | NS $\rightarrow com.$ |
| example.com | NS $\leftarrow com.$ | example.com | NS $\leftarrow com.$ |
| a.b.example.com | A $\rightarrow example.com.$ | b.example.com | NS $\rightarrow example.com.$ |
| a.b.example.com | A $\leftarrow example.com.$ | b.example.com | NS $\leftarrow example.com$ |
| | | a.b.example.com | NS $\rightarrow example.com.$ |
| | | a.b.example.com | NS $\leftarrow example.com$ |
| | | a.b.example.com | A $\rightarrow example.com.$ |
| | | a.b.example.com | A $\leftarrow example.com.$ |

Table 1: QNAME Minimisation Process [7]

In the same study from Wouter et al. [7] it is also indicated that in reality, nearly no resolver is strictly following the process as described with the RFC 7816 but implementing their own qmin style. To illustrate, some implementations always use A-record queries. So to be able to monitor qmin queries, one should check if “just the necessary portion” for the analysed DNS level is sent as a query to the NS or not.

It is essential to investigate qmin further since qname minimisation applied queries which their characteristics can be seen in Table 1 are also seen in the .nl authoritative NS. As this is a new standard and recursive resolver operators are in the transition phase to this standard, the records of qmin enabled resolvers are most probably well maintained real recursive resolvers. In the same study mentioned above, they found out that from April 2017 to October 2018, adoption grew from 0.7% to 8.8%, which is a considerably significant development for a new standard [7].

## 2.4 Machine Learning

Artur Samuel defines machine learning as follows: “machine learning gives computers the ability to learn without being explicitly programmed” [21]. As can be understandable from this simplest way of describing it, machine learning is the study of algorithms which can learn and make predictions on data. Therefore in machine learning, there are no explicit instructions, and the algorithm itself focuses on patterns and inferences. This is the main difference between machine learning models and statistical models.

Statistical models are used for creating relations between data points. With the help of statistics, data can be interpreted, but making predictions on data is not one of the strengths of statistics and statistical models even though it is possible to use statistical models for prediction purposes. In this research, the aim is to predict the source of the recursive resolvers. Thus, an approach with high capability of prediction making is needed. That is why machine learning techniques will be used for classifying the recursive resolvers.

To be able to make predictions on data, foremost, a data set formed of different features so-called a feature set is needed. This feature set is determined by selecting the most important properties which reflect the best pattern for the intended goal from the pool of different features of the data set. This is called feature engineering, which is the process of using domain knowledge of the data to create features which make machine learning algorithms work. Statistical models will be used for achieving the aforementioned needs.

Feature engineering generally considered to be is an informal topic, but it is essential in applied machine learning. In order to move my research to a real-world example, it is important to experiment if classification of recursive resolvers are possible by analysing their patterns, by using feature engineering and making decisions on the features is an inevitable part. Andrew Yan-Tak Ng said that “coming up with features is difficult, time-consuming, requires expert knowledge. Applied machine learning is basically feature engineering.” [22].

Machine learning algorithms run by constructing an example model from a training set of input values, which their outputs are pre-defined so that they can make predictions which are called outputs. These output values of the training set are often called ground truth, and it is expected to be as precise as possible to be able to build the right model on the output values.

In the last phase of machine learning, a machine learning algorithm should be selected to form the model, which is also called a classifier. Different algorithms follow different data modelling approaches, so it is important to know the dynamics of the prepared data set in order to choose the best modelling approach for the classification case.

In a survey by Nguyen et al., different techniques from different researches on internet traffic classification was analysed [23]. In the analysed papers, there were different kinds of internet traffic data which were used in classification cases. For instance, in the paper published by Roughan et al. telnet, FTP (data), kazaa, streaming, DNS and HTTPS traffics were used to create machine learning models for measurement-based classification of traffic for QoS (Quality of Service) based on statistical application signatures [24]. There are many more machine learning classification examples using different kinds of internet data as their sets in the mentioned survey.

Using machine learning on internet related data sets to solve desired classification problems is not a new approach; however, to the best of my knowledge, no one tried to classify resolver origins by their behavioural patterns. This research is aiming to follow necessary machine learning steps to create a set of features, a training set from a created ground truth and a model for the final goal of classifying the recursive resolvers of .nl ccTLD NSes.

Later this section, one of the popular machine learning algorithms: the "random forest classifier" will be inspected deeper. One of the main reasons this algorithm is popular is coming from the algorithm's randomness on building trees. This method of randomness is called random subspace method which attempts to reduce the correlation between estimators by training the estimators on random samples of features rather than all the feature set. With correctly tuned parameters, this algorithm runs effective on many different data sets [25]. Furthermore, later in this section, the importance of feature elimination to reduce the data complexity and to find the most relevant features which reflect the best pattern in the selected feature set will be discussed.

### 2.4.1 Random Forest Classifier

Random forest is a machine learning procedure to develop prediction models. It was first introduced by Breiman in 2001 [26]. It is actually an extension to Breiman's other study on bagging predictors [27]. In a simple way, we can say that random forests are a set of classification and regression trees [28]. They are simple models running binary splits on prediction variables to be able to make decisions. In that sense, they are a subset of decision trees. Random Forests can be used either for classification or a continuous response for regression. Similarly, the prediction variables can be categorical or continuous.

In a recent study from J.L.Speiser et al. [29] working mechanism of random forest classifier was explained as follows "many classification and regression trees are constructed using randomly selected training data sets and random subsets of predictor variables for modelling outcomes. Results from each tree are aggregated to give a prediction for each observation". On the other hand, drawbacks of the random forest classifier were also emphasised as "though it offers many benefits, decision tree methodology often provides poor accuracy for complex data sets". Therefore, reducing the dimensionality is important for random forest classifiers.

However, in the book called Ensemble Machine Learning by Cha Zhang and Yunqian Ma [30], benefits of random forest classifier were evaluated in two main categories: computational standpoint and statistical standpoint. In the computational point of view, it was underlined that random forest could naturally handle both regression and multi-class classification problems. It is relatively fast to train and predict, depending on only a couple of tuning parameters because it has a built-in estimate of generalisation error and can be used directly for high-dimensional problems. At the statistical standpoint, the random forest provides measures of variable importance, differential class weighting, missing value imputation, visualisation, outlier detection and unsupervised learning.

### 2.4.2 $\chi^2$ Test Analysis For Dimensionality Reduction

$\chi^2$ (chi-square) test is a statistical approach to determine the relations of features with the target set. It was first introduced by Karl Pearson in 1900 [31]. He named the test as chi-square, goodness-of-fit test because he was working on the testing of hypotheses and estimation of unknown parameters. This led to the development of statistics as a separate discipline [32].

Nowadays, the chi-square test is being used in the machine learning approaches for reducing the features -meaning dimensions- of problems. This reduction diminishes resource usage while training and labelling new data. It also decreases the over-fitting percentages of a classifier as dimensions are reduced and the problem became less complex. In contradistinction to most statistic tests, the chi-square test can provide information on the significance of observed differences, and also it provides detailed information on which specific categories explains any differences found. Therefore, because of the amount and detail of information it provides, it makes the test popular among researches [33].

In a research on probabilistic feature selection method for text classification by Uysal et al., an overview of how the test working is explained [34]. According to the paper, the chi-square test can be used for testing the independence of two events. To illustrate, the events, X and Y, are assumed to be independent if:

$P(X \cdot Y) = P(X) \cdot P(Y)$

By calculating the chi-square scores for all the features, it can be ranked the features by the chi-square scores, and then top-ranked features are selected for model training. In this paper, I will use the chi-square test to be able to rate the contribution of each feature to the target set.

## 2.5 Recursive Resolver Classification: .nz Example

.nz is the Country Code Top Level Domain (ccTLD) of New Zealand. Classifying recursive resolvers idea is inspired by from a study explained in .nz registry blog as a blog post [35]. In the study, recursive resolvers are divided into two classes, namely “monitors” and “resolvers”. The idea behind this double-split was to distinguish resolvers which are originating from monitoring tools and the resolvers which are related to end-users. To the best of my knowledge, this experiment is the closest study ever done on the classification of recursive resolvers.

To be able to achieve their goal, machine learning techniques are used. Thus, ground truth is formed to further train machine learning classifier. They collected known monitor IP addresses from ICANN monitoring, Pingdom monitoring, ThousandEyes monitoring, RIPE Atlas Probes, RIPE Atlas Anchors and AMP. They also collected known resolvers IP addresses from ISPs, Google DNS, OpenDNS, Education & Research, which they knew their queries were originating from end-users.

Next, a feature set consisting of 66 features are created in the beginning. This feature set consisted of different sections. For each source, they extracted the proportions of DNS flags, common query types, and response codes. For activity, they calculated the fraction of visible weekdays, days and hours. Next, they aggregated those by day and constructed time series for query count, unique query types, and unique query names. Then they generated features for these time series using descriptive statistics such as mean, standard deviation and percentiles.

After all, a feature selection process is followed with univariate feature selection algorithm. In the end, most relevant 50 features are selected as the final feature set. In my dataset, there are some similar features which are explained later in Section 3.4. This also shows the relevance of the selected features to the topic. However, in the .nz blog, they did not disclose each feature they have used to carry out this research.

As a result, by using automated machine learning technique with efficient Bayesian optimisation methods, their classifier reached an accuracy of 0.991 and an F1 score of 0.995. In my research, a similar approach will be followed. However, instead of classifying the resolvers as “monitoring resolvers” and “real resolvers”, their origins will be examined to be able to improve the study further.

# 3 Methodology

This section expands the data set used in this study and elaborates the selected and created feature set to conduct this research

## 3.1 Database

All the data that is used to conduct this research is provided by Stichting Internet Domeinregistratie Nederland (in English: Foundation for Internet Domain Registration Netherlands) (SIDN) which is the registry of .nl. At SIDN, all network data from two out of four NSes are collected and stored by a system called “ENTRADA”. ENTRADA is an open-source big data platform designed to ingest and quickly analyse large amounts of network data, even in a small cluster.

As a system, ENTRADA is a high-performance data streaming warehouse (DSW). ENTRADA consists of multiple components, there are generic, and ENTRADA specific components and all of them are open source. The components of the system can be seen in Figure 7.

At SIDN NSes, network traffic is collected in pcap format. To be able to provide optimisation for query lookup times, pcap files are converted into Parquet files and then stored in the Hadoop file-system (HDFS) of the Hadoop cluster. Access to these files is done by Impala which is the massively parallel processing (MPP) structured query language (SQL) query engine or any Parquet compatible engine, such as Apache Spark. Applications and services are built on top of the platform and access the platform through a variety of standardised interfaces such as SQL, Java JDBC and Python DB API [4]

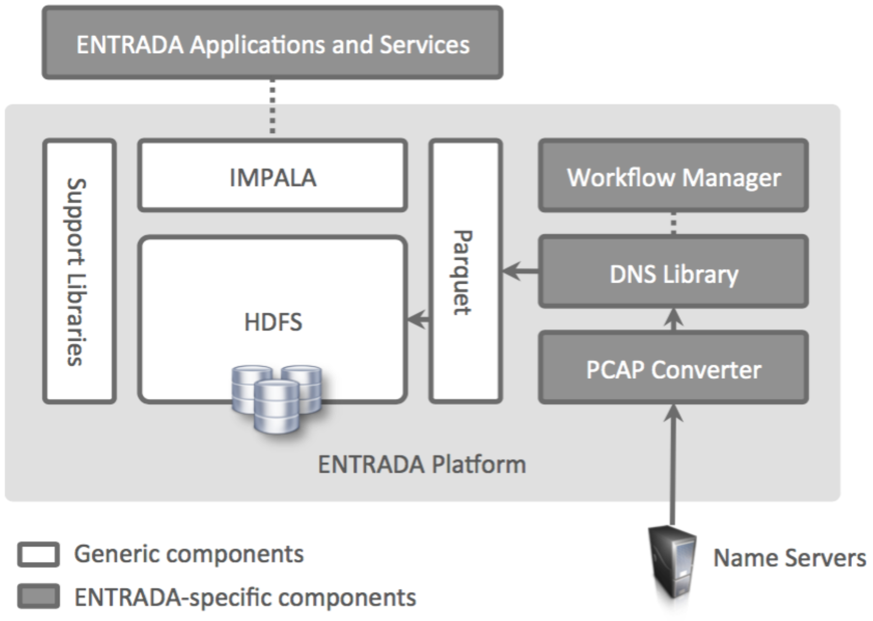


Figure 7: Components of ENTRADA [4]

In Appendix A, all columns and explanations of the database provided by SIDN to conduct this research is listed. In the database, query and reply of single a request are concatenated together in a row with 67 distinct features.

## 3.2 Ethic Concerns

While conducting this research, IP addresses of recursive resolvers are used for uniquely identifying the resolvers. IP addresses are considered to be a type of personal information. Therefore, it is important to mention that all the IP addresses are handled with care, so no specific IP address from the research is shared in this paper to ensure the privacy of IP address owners. Further information on the privacy of the data collected at SIDN, can be found in the paper called “A privacy framework for ‘DNS big data’ applications” [36]. In the paper, it is clearly stated that the only purpose of data processing in ENTRADA is to prevent frauds & abuses and enhance the stability of the .nl zone and the Internet itself.

Furthermore, there is a concern that DNS queries could reveal personal information which is mentioned in RFC 7626 [37]. In the prepared dataset, there are features that use aggregations of queries. On example is “www.” usage explained in Section 3.4.13. It is important to underline that while deriving this feature, only shares of “www.” usages are aggregated as numbers for each IP address and no full query analysis run on DNS data. This methodology is valid for other features which might be a subject of personal information. Therefore no personal information is revealed with this research.

Another essential thing to underline is about the possible misuse of the techniques of this research. Misuse of this research can lead to discriminate against users that use their own resolvers at home. One scenario on this is, the classifier that is constructed can come to the conclusion that an IP is not “important” and therefore operators of the NSes might not serve it any traffic anymore. This is clearly not the purpose of this research, and even in contrary, this research is aimed to improve the quality of service for all recursive resolvers which contacts NSes.

Any discriminatory misuse of this research is forbidden, and the operators who might do this is responsible for their own actions.

## 3.3 Ground Truth Formation

In machine learning researches, ground truth formation is one of the most important steps. It is vital because ground truth is used for teaching any machine learning algorithm how classes should be separated from one another. Furthermore, it is also used for testing the accuracy of a classifier so that one can know the percentages of misclassified instances which can also be beneficial to improve the ground truth in case classifier is classifying the instances below the desired accuracy.

The pie chart below in Figure 8, shows the companies and their traffic percentages at .nl NSes in March. The names are mapped from their autonomous system numbers. From the chart, it is clear that ISPs, large open DNS services, Cloud firms and IT-related companies are forming half of the traffic in .nl NSes. Furthermore, in the others section, there are company ASNs related to universities, hosting firms, telecommunication firms, research groups and also some probes which send queries for testing.

In my ground truth formation phase, I am expecting to see resolvers more or less coming from the aforementioned sector origins. However, IP addresses originating from these sectors are going to be selected on some conditions to make sure the accuracy of the classes is as improved as possible.

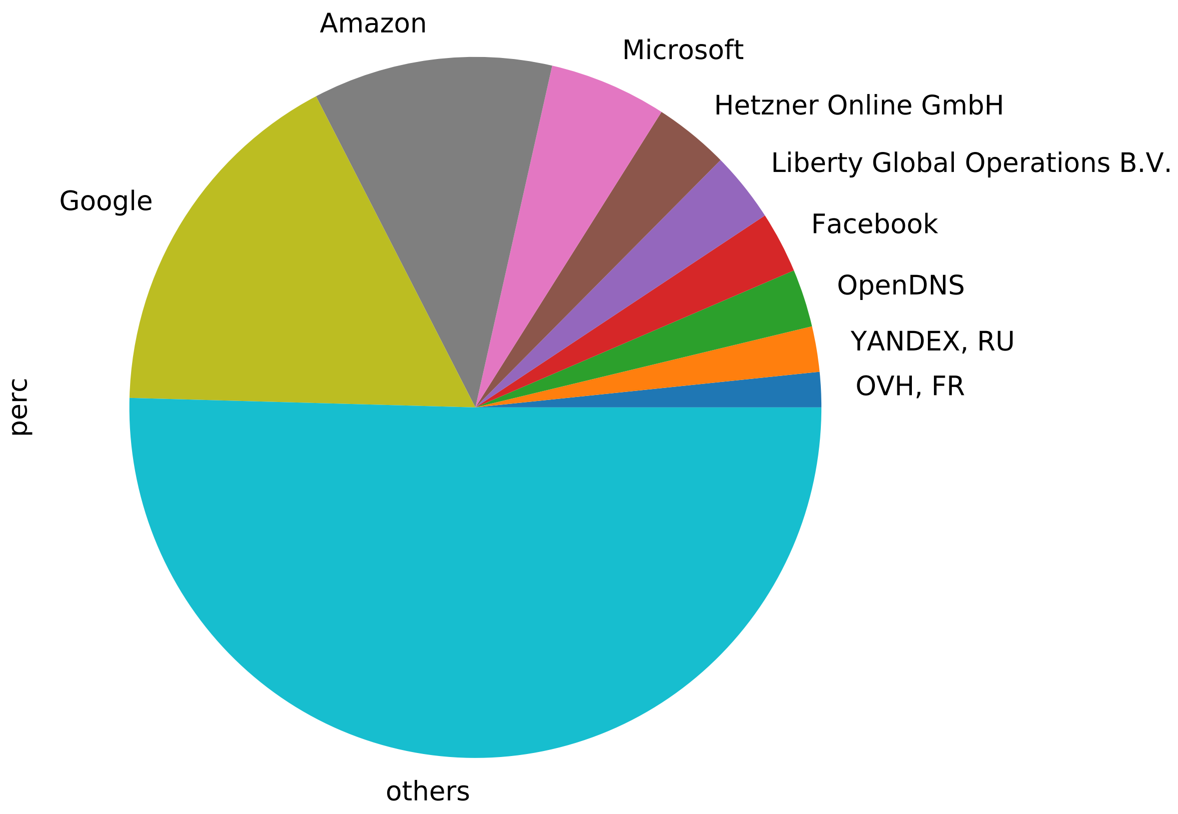


Figure 8: Companies and their traffic percentages at .nl NSes in March

In SIDN Labs (the research group of SIDN), for research purposes, specific query data is collected from Réseaux IP Européens (RIPE) probes and Luminati Proxy Service clients and stored in ENTRADA. In the following subsections, how these measurements are used for creating the ground truth will be explained together with other sources that are also used for this classification.

### 3.3.1 Luminati Proxy Service Data & RIPE Atlas Measurements Analysis

A Virtual Private Network (VPN) service acts as a direct connection, allowing clients to send all of their information from a virtual link. Therefore, all requests are sent to a VPN server, and this server then forwards clients' request to the target website by using a different IP address. This guarantees clients' privacy as the website only see the information of the IP address they gathered from the request, thus keeping clients' location anonymous. The Luminati Proxy Service is similar to a VPN in the way information is transferred, but instead of sending clients' request from a different IP, it connects a client to a network of thousands or even millions of alternative IP addresses where every client is a potential exit node [38].

In SIDN, to gather the measurements from Luminati, the operators of Luminati was paid for access, and their terms of service were held. This method of data collection is similar to the case in a research where Luminati Proxy Service was used to collect needed measurements [39]. Furthermore, the owners of exit nodes agreed to route Luminati traffic through their hosts in exchange for free service.

From Figure 9, it can be seen the environment used for collecting the queries. To form this database, queries are sent from all Luminati clients to a domain under our control once per day and every day with different unique numbers. Thereby, the upstream IP addresses of recursive resolvers that are used by these clients are collected along with some other fields of the query packet, which can be seen in Appendix B.

For collecting these queries, an NS is built to be the authoritative NS of a second-level domain. In this paper, this authoritative NS is represented by example.nl's NS, but the real domain name used for this research is still being used for the same and other research purposes, so it will be kept confidential to prevent unnecessary traffic to it. It can be assumed that example.nl's authoritative NS traffic is observable to the experiment conductor, which is SIDN Labs in this case.

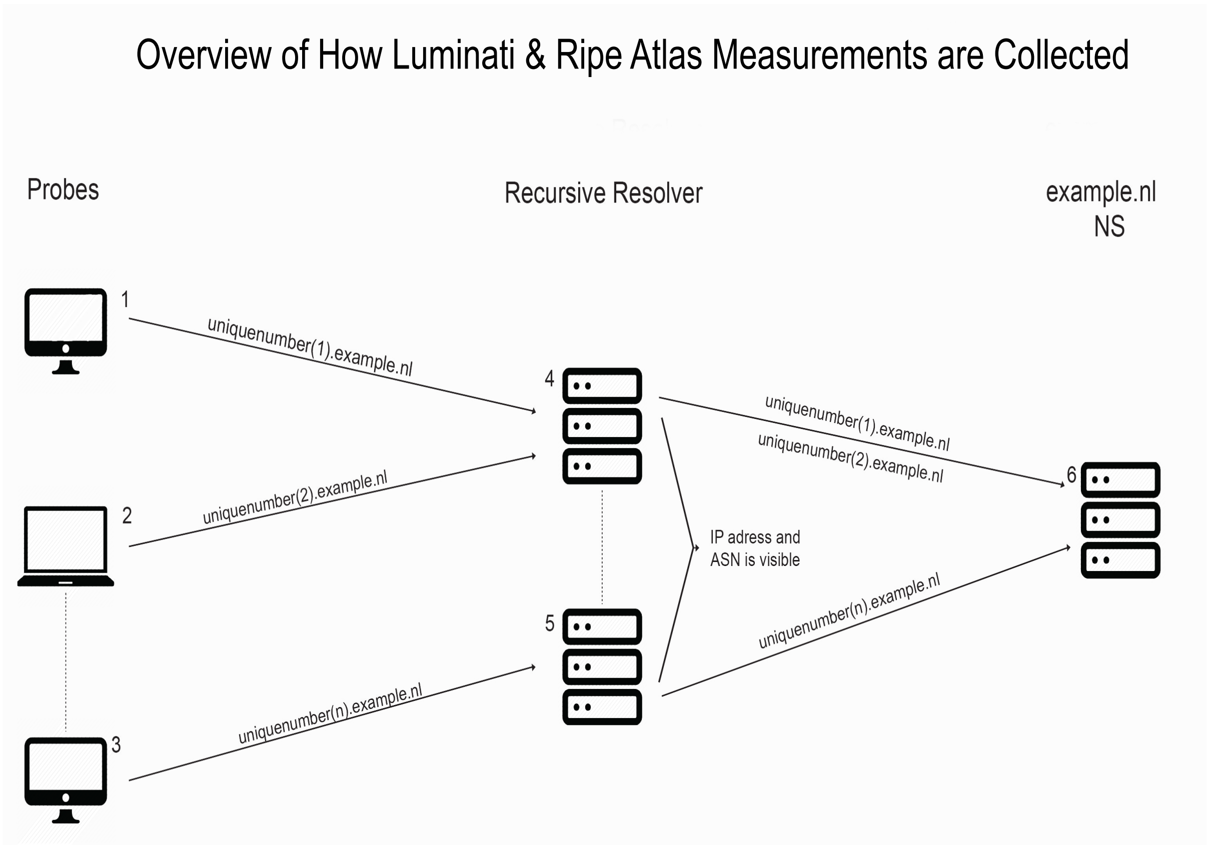

Overview of How Luminati & Ripe Atlas Measurements are Collected
Probes
Recursive Resolver
example.nl NS
1
uniquenumber(1).example.nl
2
uniquenumber(2).example.nl
3
uniquenumber(n).example.nl
4
5
6
uniquenumber(1).example.nl
uniquenumber(2).example.nl
IP adress and ASN is visible
uniquenumber(n).example.nl

Figure 9: Overview of how Luminati & Ripe Atlas measurements are collected

For each probe in the Figure 9 represented by numbers 1,2 and 3, a unique query was created so that regardless of which recursive resolver contacted to the authoritative NS for name resolution, it is known if the query is originating from a distinct host. Thereby, even though two or more distinct clients are using the same recursive resolver for name resolution, which is the case represented by number 4 in the Figure 9, it can be said with a full certainty that those queries are initiated from two (or more) distinct clients even though the IP address and Autonomous System Number (ASN) contacted to the authoritative NS is the same. There can also be some IP addresses seen in the authoritative NS which has only one query during the day, which is represented by number 5.

The number 6 is the final destination of the query before the answer is returned to the client for the name resolution by the recursive resolver. The assumption is that, if many queries from different clients are coming from same recursive resolvers, then those resolvers are likely to be popular recursive resolvers serving to their clients.

By having unique numbers for clients, it is guaranteed that nothing is cached and that every client can be identified. This also means that, if 1000 unique client numbers are observed from the same IP in a day, then with no doubt it is possible to say that the resolver served at least 1000 different clients.

The data set to be fed to the machine learning algorithms is created on day March 20, 2019 as mentioned in Section 3.4. Therefore, while creating the ground truth, Luminati measurements run in March are used.

To start, from Luminati measurements in ENTRADA, all ASNs are collected throughout March which are originated from The Netherlands, Germany and Belgium because I expect most user-generated traffic from these countries. As the next step, these ASNs are labelled one by one to their company names and sectors that they are working on. From this manual mapping, seven different sectors are observed which are listed below:

- ***Cloud Firms***: Companies which provides resources like data storage and computing power to their clients online.
- ***Hosting Firms***: Companies which provides web hosting for their clients.
- ***Internet Service Providers (ISPs)***: Companies which provides services for accessing, using, or participating in the Internet to their clients.
- ***Information Technology Companies (IT Firms)***: Companies which are working in the information technology field providing solutions on software, hardware and so on to their clients.
- ***Research Related ASNs***: Universities, research-oriented foundations and so on.
- ***Telecommunication Companies***: Companies which provides a collection of nodes and links to enable telecommunication for their clients by using electrical signals or electromagnetic waves
- ***Open Resolvers***: Companies which are willing to resolve recursive DNS lookups for anyone on the internet.

ASN to sector mapping can be observed in Appendix C. Along with the ASNs observed from RIPE measurements are concatenated together.

From the ASNs observed in whole March, from Luminati measurements, all distinct IP addresses are extracted from the database which had at least 100 distinct unique query number and labelled as the same sector as its ASN. The threshold of hundred is chosen because these measurements are run every day once and if an IP address has 100 unique query numbers, it means that roughly every day, that IP address served at least three clients. It is thereby eliminating the unpopular recursive resolvers in March. In total, 3131 IPs were found and added to the ground truth list along with their class labels.

RIPE Atlas is a probes network which is measuring the connectivity and reachability of the Internet [40] therefore allowing observation of the Internet in real-time. There are thousands of active probes in the RIPE Atlas network, and this number is growing continuously. The RIPE NCC collects data from the aforementioned network of probes to provide aggregated results for different purposes. RIPE Atlas users can make use of these aggregated data. Furthermore, users who also host a probe can use the entire RIPE Atlas network to conduct customised measurements.

SIDN Labs also hosts a virtual RIPE Atlas probe to help one of the biggest research network on observing the situation of the Internet. As mentioned in their page, anyone who hosts a RIPE Atlas probe can conduct their own customised measurements in order to gain valuable information about their network using other RIPE Atlas probes [40].

Similar to what has been done in Luminati measurements, a unique query per Atlas probe per day is sent to a domain name which its authoritative NS is in SIDN's control.

Similar to Limunati measurements, also in RIPE Atlas measurements, the data is collected by giving a unique number to each Atlas probe represented as a probe in Figure 9. To start, from RIPE Atlas measurements in ENTRADA, all ASNs are collected throughout March which are originated from The Netherlands, Germany and Belgium. This returned 54 distinct ASNs of which 4 of them were different from Luminati measurements. Next, these ASNs are labelled to their company names and sectors that they are working on. From this manual mapping, six different sectors are observed which are listed below:

- ***Cloud Firms***: Companies which provides resources like data storage and computing power to their clients online.
- ***Hosting Firms***: Companies which provides web hosting for their clients.
- ***Internet Service Providers (ISPs)***: Companies which provides services for accessing, using, or participating in the Internet to their clients.
- ***Information Technology Companies (IT Firms)***: Companies which are working in the information technology field providing solutions on software, hardware and so on to their clients.
- ***Research Related ASNs***: Universities, research-oriented foundations and so on.

- ***Open Resolvers***: Companies which are willing to resolve recursive DNS lookups for anyone on the internet.

ASN to sector mapping can be observed in Appendix C along with the ASNs from Limunati measurements concatenated together.

From the ASNs in March from RIPE Atlas measurements, all distinct IP addresses are extracted from the database which had at least 100 distinct unique query number and labelled as the same sector as its ASN. The threshold of a hundred is chosen because of the same reasoning as Limunati measurements. So if an IP address has a 100 unique query numbers, it means that roughly every day, that IP address served at least three clients. It is thereby eliminating the unpopular recursive resolvers in March. A total number of 1920 IPs were found. Only 368 of them were different from Limunati measurements. Non-conflicting IP addresses are also added to ground truth list along with their class labels.

### 3.3.2 Open Resolvers (Large Public DNS Services) List

A DNS resolver is called an open resolver if it provides recursive name resolution for its clients outside of its administrative domain [41]. There are lots of open resolvers working to serve their clients. This research is about predicting the origin of the resolvers from name resolution behaviours of them. Therefore I'm looking into major public DNS providers because they represent a large user base, which is relevant for operators. In the following subsections, it will be explained how the data is created for ground truth.

#### 3.3.2.1 OpenDNS Resolvers

As described in their web page: "OpenDNS was founded in 2006 with the mission to provide a safer, faster, and better internet browsing experience for all users. Since then, OpenDNS provided a recursive DNS service for use at home, and in 2009 introduced a service for the enterprise market." [42].

The resolver list of OpenDNS is publicly available on their website [43] and this is how the IP addresses from OpenDNS Resolvers are included in the ground truth data set.

To be able to create a list of the IP addresses which contacted .nl NSes on day March 20, 2019 and belongs to OpenDNS resolvers, IP addresses are selected which had a match with resolver list of OpenDNS. A total number of 722 IP addresses were added to the ground truth by this method.

#### 3.3.2.2 Google Public DNS Resolvers

Google explains its public DNS service as: "Google Public DNS is a recursive DNS resolver, similar to other publicly available services. It provides many benefits, including improved security, fast performance, and more valid results." [44].

Similar to OpenDNS resolvers, IP addresses which are originating from Google Public DNS resolvers are found by querying TXT records of Google Public DNS as “dig TXT locations.publicdns.goog.”

From the list of IP addresses from Google Public DNS, a ground truth is created by matching IP list with the IP addresses seen in .nl NSes on day March 20, 2019. By this way, a total number of 2792 IP addresses were added to the ground truth list.

#### 3.3.2.3 Quad9 Resolvers

According to Quad9’s web site: “Quad9 is a free, recursive, anycast DNS platform that provides end users robust security protections, high-performance, and privacy.” [45].

While conducting this research, a blog post was written to be published in Asia-Pacific Network Information Centre (APNIC)’s blog. From that blog post, operators of Quad9 resolvers are contacted [46]. From there, a list of egress IP addresses of Quad9 was obtained. Unfortunately, these IP addresses are asked to be kept confidential, so I will not be able to share the list of IP addresses or ASNs of Quad9 recursive resolvers. In total, 32716 IP addresses were added to the ground truth from Quad9 resolver list.

After getting the IP addresses of Quad9 resolvers, all IPs on day March 20, 2019 from .nl NSes are searched, and IP addresses which matched with Quad9 resolvers’ IP addresses are kept in another list to be further used in the classification as a ground truth.

### 3.3.3 Combining Ground Truth Data Sets Together

When ground truth formation ended for each distinct analysis type, there were multiple data sets which included IP addresses of recursive resolvers and which were classified according to their serving sectors.

All IP addresses of recursive resolvers which are assigned to one of the seven different classes are given an integer value for the classification algorithms to understand them.

While combining these data sets together, there were 137 IP addresses which were manually classified as originating from ISP resolvers according to their ASNs. These IPs were coming from the Limunati & RIPE measurements, and those IPs were also included in the IP list of the open resolvers. I chose to label them in the open resolver class because the IP addresses which are labelled as open resolvers are belonging to that class with higher accuracy. Limunati and RIPE measurements are mapped to their sectors manually, and so there might be some misclassified instances, or an instance might be missing a sector that the company is active on as well.

By this way, these classes are transformed into something which any classification algorithm in Python’s scikit-learn library can process [47].

In the end, there were 39,361 distinct IP addresses each representing a recursive resolver in the ground truth data set with their target classes as integers between [0,6].

## 3.4 Data Set Creation for Machine Learning

In researches, which applies machine learning, deciding which features might be useful for the solution of the defined problem is one of the most critical parts of the research. Therefore, it is important to select these features carefully by running experiments to see the behaviour of each feature. Thereby, one can produce reasoning on why that specific feature might be useful in solving that certain problem.

In the provided DNS traffic database, with using any combination of the available 67 distinct features, one can extract any desired information by querying the database with SQL. In this study, as a beginning, shares of seventeen features from the DNS database are included directly to the created data set, and ten features are derived from examining other fields of the DNS database. A total number of 27 features are created to be able to test the performance. These 27 features are derived from the domain knowledge that I gained from the previous section, the literature review.

In this research, to be able to conduct behaviour analysis on IP addresses to decide which features to include to the research, the day of March 20 2019 was selected randomly. All the analysis in the following subsections is made on IP addresses which reached to .nl NSes on the aforementioned day. Later on, to be able to validate the classifier, another day from May will be selected to create another data set on the selected features and will be tested with the classifier trained on March 20, 2019.

### 3.4.1 Response Code Field

Response Code (RCODE) is a 4-bit field which set as the part of responses to queries. This field shows the success of a query.

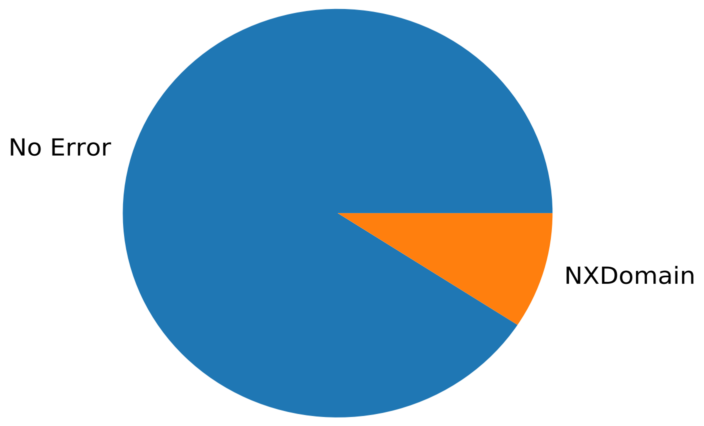


Figure 10: RCODE percentages on day March 20, 2019

From the Figure 10, it can be seen that 90% of all queries return an rcode of 0

(No Error) and 9% of them returns an rcode of 3 (NXDomain). All the other kinds of response codes share 1% of daily traffic.

#### 3.4.1.1 No Error Share

In RFC1035, the explanation of RCODE when set to 0 is “No error condition” [9]. No Error responses form 90% of all 1.2 billion queries seen on March 20, 2019. It is important to use the shares of this feature by IP addresses in the created dataset since it shows the successful query percentages of IP addresses. This feature can be beneficial for classification in terms of categorical behaviour of successful queries.

To be able to normalise the data for further usage in classification cases, for each IP address, the count of RCODE=0 is divided into the total number of queries from that IP address.

#### 3.4.1.2 Name Error Share

From RFC1035, Name error means that the domain name referenced in the query does not exist. This answer is meaningful only for responses from an authoritative name servers [9]. This field can be beneficial in explaining a category which has rather higher shares of its queries as Non-Existent Domain.

To be able to normalise the data for further usage in classification cases, for each IP address, the count of RCODE=3 is divided into the total number of queries from that IP address.

### 3.4.2 Resource Record Types

Resource Record (RR) define data types in the Domain Name System (DNS). Resource Records identified by RFC 1035 are stored in binary format internally for use by DNS software [9, 48]. To be able to choose which RR types to be included in the feature set, a popularity analysis of RR types is done as a first step.

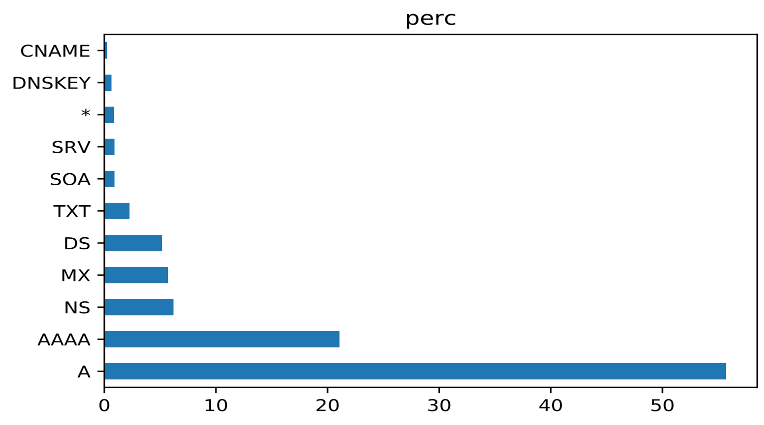


Figure 11: Top 11 RR type percentages on day March 20, 2019

In Figure 11 percentages of each RR type on day March 20, 2019 can be seen. In .nl NSes, most popular RR types are A, AAAA (quad-A) and NS records

#### 3.4.2.1 A Record Share

When a resolver sends a query to an NS which its query type field is set to 1, an A record entry is searched in NS for the desired domain name which is indicated in the query and returned an IPv4 (IP Version 4) address if it exists. It is not surprising that this type of RR is still the most queried RR type since more than half of the internet is still working on IPv4 addresses according to the latest report on “state of IPv6 deployment in 2018” published by Internet Society[49].

The most popular RR type received at .nl NSes is A record. However, .nl NSes are not authoritative for A records of second-level domains. In a simpler way, if a resolver requests for an A record of example.nl, it will receive the NS record of example.nl. There are some resolvers which their settings are advanced so that they do not send queries to NSes which they know those NSes are not authoritative to what they are looking for. Nonetheless, as it can be seen from the statistics of Figure 11 shares of RR types which .nl NSes are authoritative to (*e.g.* NS records DS records and so on) are not high. This is an indirect way to prove that mentioned resolvers are not forming the majority of .nl NSes.

Similar to the RCODE field, to be able to use the data further in the classification share of A records for each IP address is normalised. The shares are calculated for each IP address by dividing the total number of A record queries to the total number of queries from that IP address on March 20, 2019.

#### 3.4.2.2 AAAA Record Share

AAAA (quad-A) record is the second most queried RR type in .nl ccTLD NSes. When a resolver sends a query to an NS which its query type field is set to 28, a quad-A record entry is searched in the NS for the desired domain name which is indicated in the query and returned an IPv6 (IP Version 6) address if it exists [50].

As aforementioned in A Record section, .nl NSes are not authoritative for quad-A records of second-level domains. Instead, they are authoritative to provide authoritative NSes for the desired second-level domains of .nl namespace.

From the Figure 12, it can be seen that in .nl NSes on March 20, 2019 when 80% of all resolvers are selected, roughly 50% of queries are asking for AAAA queries. AAAA records are popular for end-users to query as AAAA asking queries are mapping a domain name to its IPv6 address. Therefore it can be useful in labelling ISP and Open Resolvers in the dataset.

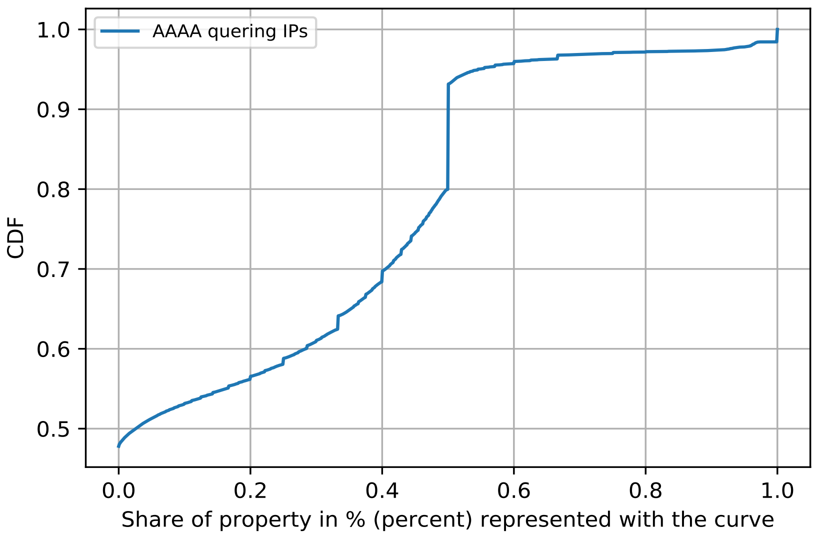


Figure 12: CDF Graph of shares of IP addresses on AAAA RR type on day March 20, 2019

Similar to previous RRs types, shares of quad-A are calculated by dividing the total number of quad-A record queries to the total number of queries from each unique IP address on day March 20, 2019.

#### 3.4.2.3 NS Record Share

NS records delegates a DNS zone to use the given authoritative name servers. This record type is the record type which .nl ccTLD NSes are authoritative to. This RR type is thought to be useful in distinguishing recursive resolvers which are level-aware in terms of domain name levels (eg., top-level domain, second-level domain) so that they are querying for 'correct' type of RR for a ccTLD, the NS records. In addition, there can be resolvers querying for NS records to refresh their cashes when time to live value of a record expires.

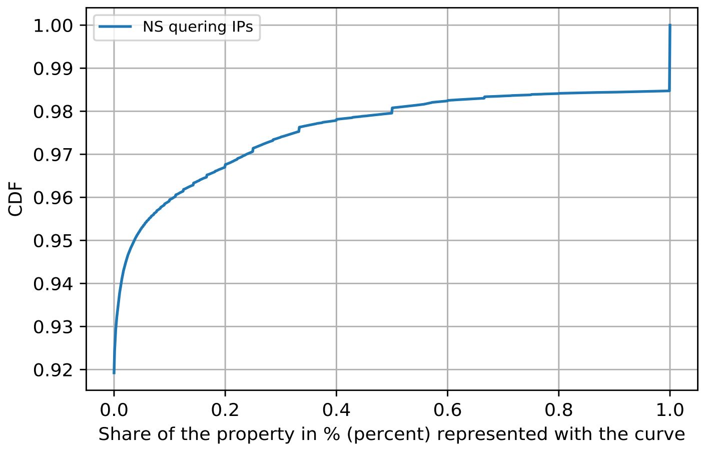


Figure 13: CDF Graph of shares of IP addresses on NS RR type on day March 20, 2019

In .nl NSes on day March 20, 2019 it can not be seen any NS queries until 92% of all IPs are selected. When 97% of all resolvers are selected, roughly 25% of their queries are asking for NS records.

Shares of NS records are calculated for each IP by dividing the total number of NS record queries to the total number of queries from that specific IP address on day March 20, 2019.

#### 3.4.2.4 CNAME Record Share

Canonical Name (CNAME) is a type of RR in DNS which maps one domain name to another one, in this case to a Canonical Name [9]. CNAMEs do all the re-directions in the DNS. Although this type of record is the least popular one in the top 11 list in Figure 11, it is included to be able to measure the contribution ratio of it to the classifier.

CNAMEs are widely used by content delivery networks (CDNs) like Akamai, and they are mostly used by end-users. Thereby, it is expected to distinguish the ISP and Open Resolver classes from the other classes which are highly related to end-users.

Shares of CNAMEs are calculated by dividing the total number of CNAME record queries to the total number of queries from each unique IP address on day March 20, 2019.

#### 3.4.2.5 SOA Record Share

At the top level of a domain, the name database must contain a Start of Authority (SOA) record. This SOA record identifies what the best source of information for data within the domain is. SOA contains the current version of the DNS database and various other parameters that define the behaviour of a particular DNS server [48]. This record is important for Domainers. Domainers are organisations that are interested in buying domain names as soon as they become available. They could check this with an SOA query if the serial number of the zone file has increased. This could be a sign for them that new domain names are available. This record type is expected to be seen in IT Firms, ISPs and Research classes.

Shares of SOA records are calculated by dividing the total number of SOA record queries to the total number of queries from each unique IP address on day March 20, 2019.

#### 3.4.2.6 MX Record Share

Mail Exchance (MX) maps a domain name to a list of message transfer agents for that domain. A zone can have one or more MX records. These records point to hosts that accept mail messages on behalf of the host. MX record is the $4^{th}$ most queried RR type in .nl ccTLD NSes. It is expected to see this query type mostly at resolvers of Hosting Companies, IT Firms and ISPs which mailing is highly popular in end users of those classes.

Shares of MX records are calculated by dividing the total number of MX record queries to the total number of queries from each unique IP address on day March 20, 2019.

#### 3.4.2.7 TXT Record Share

Text Strings (TXT) are originally created for arbitrary human-readable text in DNS records. Since the 1990s, this has changed, and now this record more often carries machine-readable data, specified by RFC 1464, opportunistic encryption, Sender Policy Framework. Besides, TXT records are also used for DomainKeys Identified Mail and Domain-Based Message Authentication Reporting and Conformance, which are added security and authentication mechanisms. This feature can help to distinguish the resolvers who are using these machine-readable data, namely Hosting Companies, IT Firms and ISPs as they are also serving to clients who use e-mail services [51, 52].

Shares of TXT records are calculated by dividing the total number of TXT record queries to the total number of queries from each unique IP address on day March 20, 2019.

#### 3.4.2.8 SRV Record Share

Service locator (SRV) allows a service to be linked with a hostname. Any application that needs to find the location of a specific service will initiate a query for the corresponding SRV record which describes the service.

Before SRV field was invented in RFC 2782, the situation used to be like this: one must either know the exact address of a server to contact it or broadcast a question. However, after SRV RR type was invented, it allowed administrators to use multiple servers for a single domain to move services from host to host and designate some hosts as primary servers for a service and others as backups [53].

Nowadays, this record type is commonly used for Lightweight Directory Access Protocol (LDAP) and gaming purposes. Therefore it is expected to see slightly higher shares of SRV records in ISPs, Telecommunication Firms and Open Resolvers.

Similar to previous RRs types, shares of SRV records are calculated by dividing the total number of SRV record queries to the total number of queries from each unique IP address on day March 20, 2019.

#### 3.4.2.9 DS Record Share

In RFC 4034, the Delegation Signer (DS) is explained like this: “DS refers to a DNSKEY RR and is used in the DNS DNSKEY authentication process. A DS RR refers to a DNSKEY RR by storing the key tag, algorithm number, and a digest of the DNSKEY RR” [54].

As described in Section 2.2.2, DNSSEC validating resolvers use DS records to verify the authenticity of child zones. Therefore, resolvers that query for DS records might be validating resolvers. Resolvers from Quad9 and Google do DNSSEC validation, and therefore, the DS record share might be a useful feature to distinguish Open Resolvers.

This field is essential to add to the feature set because it can be useful distinguishing security-aware a class. Similar to other RR types, the share of DS Record type was calculated for each distinct IP address on day March 20, 2019 by dividing the total number of DS queries to the total number of queries.

#### 3.4.2.10 RRSIG Share

DNSSEC uses public-key cryptography to sign and authenticate DNS RR sets. Those digital signatures are stored in RRSIG RRs. Although this field was not a member of top 11 most queried RR types list showed in Figure 11, this feature also reveals information if a recursive resolver has its settings on security and yet again can be useful to distinguish a class of recursive resolvers which has significant shares of security-enabled recursive resolvers [54].

Similar to other RR types, the share of RRSIG Record type was calculated for each distinct IP address on day March 20, 2019 by dividing the total number of RRSIG queries to the total number of queries from that IP address.

#### 3.4.2.11 DNSKEY Share

This RR type is also introduced for DNSSEC extension and mentioned in RFC 4034 [54]. The purpose of this RR is to store the DNS public keys.

Again this section can give hints to classifier for distinguishing the security-aware class. The share of DNSKEY was also obtained by dividing the total number of DNSKEY queries to the total number of queries for each resolver on day March 20, 2019.

### 3.4.3 Extension Mechanisms for DNS - DO Bit

Initially, when DNSSEC was deployed, the significant portion of queries was from recursive resolvers that were not DNSSEC aware, so they did not understand or support the DNSSEC security RR types. When a query from such a resolver is received in a DNSSEC enabled NS, as the DNSSEC specification indicates that the NS must respond with the appropriate DNSSEC security RRs, in the preliminary and experimental deployment of DNSSEC, there were reports of DNSSEC unaware resolvers being unable to handle responses which contain DNSSEC security RRs. This resulted in the resolver failing in the worst case or entire responses being ignored in the better case [55].

This is the reason that EDNS - DO bit was introduced and by this way if the DO bit in an initiating query is not set, the name server side omits any DNSSEC RRs from the response unless initiating query explicitly requested for those RR types [54].

This field is essential to add to the feature set so that classifier can find a pattern between class types and DNSSEC-awareness of the recursive resolvers. The shares of recursive resolvers which set EDNS-DO flag are calculated by dividing the total number of EDNS-DO set queries to the total number of queries coming from each distinct IP address on day March 20, 2019.

### 3.4.4 Checking Disabled

The Checking Disabled (CD) bit was introduced in order to allow a security-aware recursive resolver to disable signature validation in a security-aware name server's processing of a particular query. If the CD bit is set, it indicates that the originating resolver is willing to perform whatever authentication its local policy requires. Therefore, NS need not perform authentication on the RRsets in the response. That is, by setting the CD bit, the originating resolver has indicated that it takes responsibility for performing its own authentication, and the recursive name server should not interfere [54].

From the explanation cited from RFC 4034, it would not be wrong to interpret that if CD flag is set, that also is a sign that the recursive resolver is security-aware and wants to perform all the integrity checks by itself. This can also contribute distinguishing the classes in the classification problem. Therefore, the share of CD Flag set queries are calculated by dividing the total number of CD flag set queries to the total number of queries for each resolver by their IP addresses on day March 20, 2019.

### 3.4.5 Authoritative Answer

Authoritative Answer (AA) field is a part of the response which specifies that the responding name server is an authority for the domain name in the question section. In Figure 14 it can clearly be seen that approximately 85% of all queries get a non-authoritative answer as they are directly querying the .nl ccTLD with their desired final RR types and not taking into account for which RR types the connected server is authoritative to as earlier discussed in section 3.4.2.1.

On the other hand 14% of all queries received on day March 20, 2019 got a response of AA. It is expected that shares of AA flag set to “True” can help better explaining resolvers which are DNS levels aware and resolvers which validates the answers received from NSes with DNSSEC. That is why shares of AA flag set to “True” added to the features list by dividing the total number of AA flag set queries from each distinct resolver to the total number of queries from each resolver on day March 20, 2019.

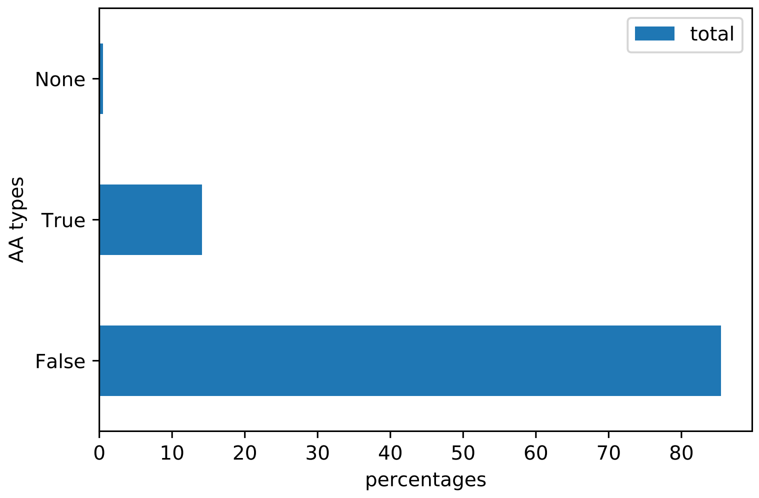


Figure 14: Percentages of Authoritative Answer bit on day March 20, 2019

For the rest of the queries, which forms less than 1% of all daily queries, either AA flag is not set to one or zero according to an error, or there was a mistake in the connection that the reply was never created, so AA section is not set or reset.

### 3.4.6 Recursion Desired

As explained in the standard resolver behaviour section in 2.1, RD field should only be set between a client and recursive resolver. Whenever a recursive resolver receives a query for any RR type which its RD flag is set, it recursively surfs in the levels of DNS hierarchy and returns the final result to the client. However, it is not expected to see a query which its RD flag is set in an NS. If there are queries in an NS with RD flag is set, it can be useful to distinguish a class of resolvers which are not correctly configured.

For this purpose, shares of RD flag set queries are added by dividing the total number of RD flag set queries to the total number of queries for each IP address on day March 20, 2019.

### 3.4.7 Query Name Minimisation

Since RFC 7816 was published in March 2016, there are recursive resolvers in the wild that is implementing query name minimisation (qmin) to their resolving algorithm, as mentioned in Section 2.3.3. From the domain knowledge, I assume that if a recursive resolver is doing qname minimisation, there is a significant probability that it is a well-maintained resolver. Therefore, recursive resolvers which belong to ISPs and Open Resolvers are more likely to implement that to their machines working as recursive resolvers.

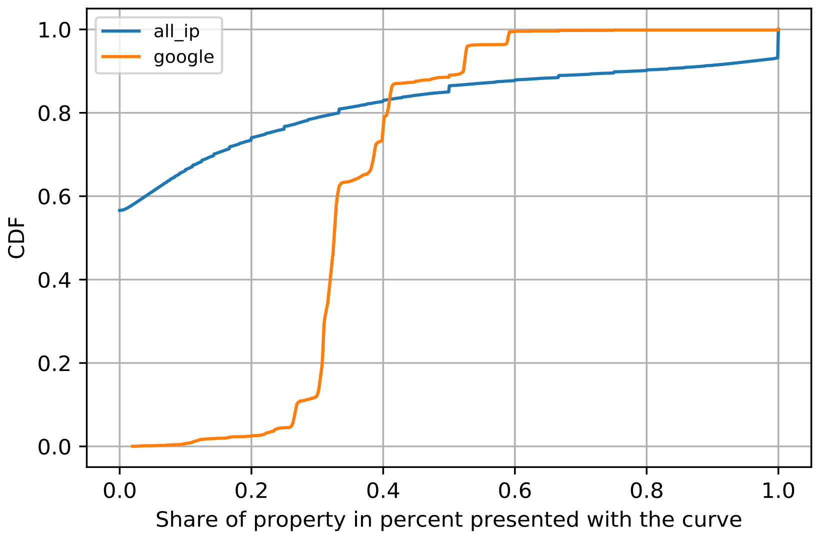


Figure 15: CDF Graph of shares of IP addresses compliant to qmin on day March 20, 2019

From the CDF graph in Figure 15 we can tell that nearly 60% of all recursive resolvers do not send any query to .nl NSes which is qmin compliant. On the other hand, if we look at the behaviours of IP addresses from Google's open resolvers, we can clearly see that if we pick 20% of all IP addresses, roughly 35% of the queries are qmin compliant. This graph also supports the belief behind well maintained recursive resolvers are more likely to implement new technologies and this feature can be beneficial to the classification process.

Shares of qmin compliant resolvers are calculated by dividing qmin compliant query numbers to the total number of queries from each distinct IP address on day March 20, 2019

#### 3.4.8 Domain Name Cover

This is a derived feature using the question section in the query. From the Appendix A one can see that in DNS queries database in ENTRADA, there are two fields for question section, "qname" which stands for query name which includes the full text received from recursive resolver and there is the "domainname" section which is the minimum required information for .nl NSes to reply the query. By counting the distinct successful queries (successful here means RCODE=0) from each resolver and dividing this number to the total number of registered domain names from the day that experiment is run, it is possible to map the percentages of domain name covers of each unique IP address. The purpose was to determine for each resolver (according to their IP addresses) how many unique domain names they are querying for a given day in this case on day March 20, 2019.

The results are as follows: The IP address which queried the most number of unique successful domain names was covering 41% of all registered domain names for that day. Another important result from this feature was that IP addresses which covered the most of the .nl domain were originating from Internet Service Provider (ISP) IP addresses, research centre IP addresses and hosting company IP addresses. This has encouraged the thought that this derived property's shares might be useful in explaining the data better in the classification case so this feature was also included to the primary dataset for later experiments on feature importance.

### 3.4.9 Port Number Deviations

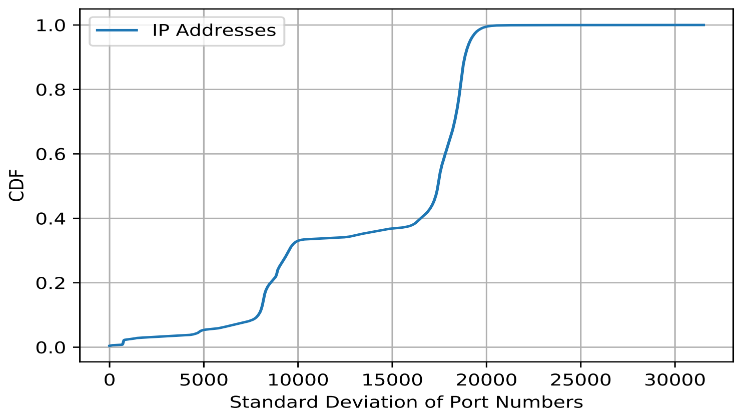


Figure 16: CDF Graph of standard deviations of IP addresses on day March 20, 2019

For every query, it is expected for a recursive resolver to initiate the connection from its random port. Thus, it is expected to observe significant differences in source ports each time same IP address queries the NS. A higher standard deviation generally means newer resolver software. This is why this feature could be relevant to create a pattern for the classifier. Controlling this by getting standard deviations of each distinct resolvers' source port number led to the graph, which can be seen in 16.

In statistics, standard deviation measures the variation of data points from its mean. Accordingly, low standard deviation shows that the data points are spread in a low range of values, and high standard deviation shows that the data points are spread to a high range of values [56]. From the figure, it can be seen that when roughly 40% of all IP addresses are selected, their standard deviations are approximately 17500. From the increasing points in the graph, we can also tell that there are two clusters and there are two linear-like plateaus which represent the distance between clusters. This will be shown with the next figure.

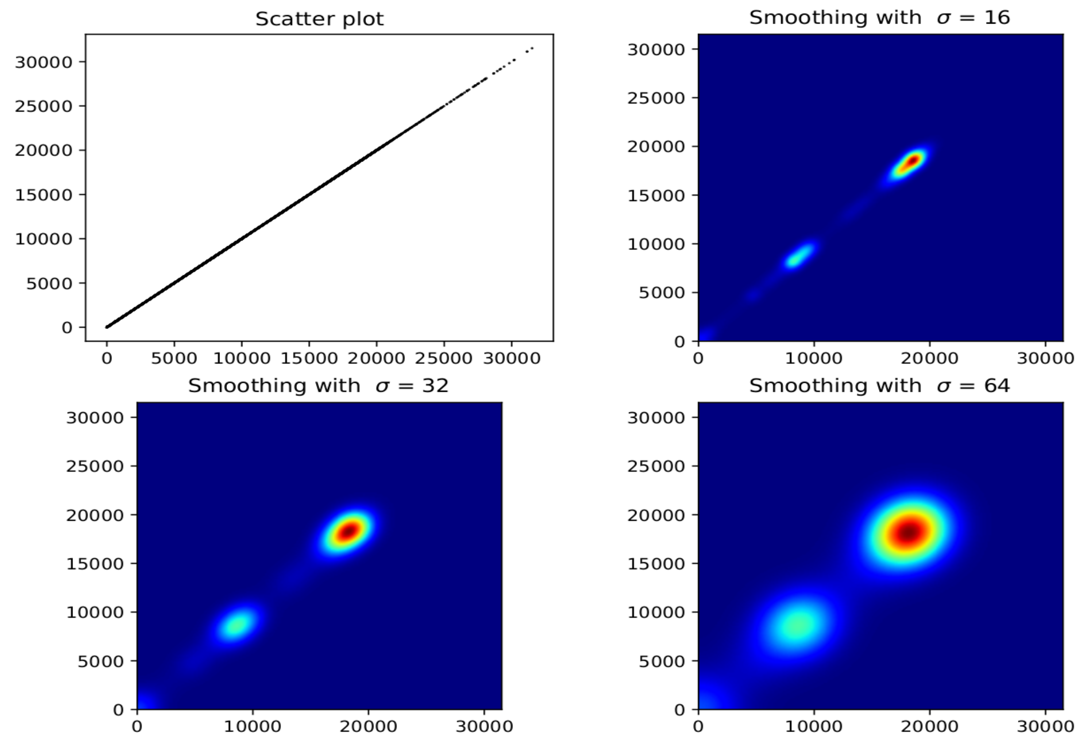


Figure 17: Gaussian Filtered 2D Histogram for Port Number's Standard Deviations

From the Figure 17 it can be seen that beginning from (0,0) there are two different clusters. The cluster between coordinates (1500,1500) and (2500,2500) is where most of the data from the dataset are concentrated. Especially smoothing with $\sigma$ = 32 shows these clusters well enough. The first cluster which its centre is around point (10000,10000), shows the resolvers which have small deviations in their source port numbers. On the other hand, the second cluster which its centre is around point (20000,20000), shows the resolvers which have high deviations in their source port numbers.

To be able to use this feature in the classification, standard deviations of distinct IP addresses are normalised by min-max normalisation between 0 and 1.

### 3.4.10 Preferred Name Server

As discussed in Section 2.3.1, all the NSes are visible to recursive resolvers, and they query nearly all of them if they do not get a reply in a preferred time scale. The assumption is that "normal" behaving resolvers do not send all their queries to just one name server. The work done by [2] inspired to look at the preferred NS choice of recursive resolvers. ISP resolvers and open resolvers are generally well maintained, and they generally show standard behaviours. On the other hand, research resolvers might not show standard behaviours. This is why this feature might help to distinguish these classes.

To be able to add this into the feature set, a logical approach was followed. From ENTRADA, for each distinct IP address on March 20, 2019, counts of queries are aggregated both for NS1 and NS2 of .nl NSes. Then the most queried NS's share which is calculated by dividing the number of most queried NS's query count to the total queries from that IP address is added to the preferred name server feature. So in the end, for all IP addresses, all the percentages were between 0.5 and 1. By this way, for each IP address percentages of most queried NS feature was created.

### 3.4.11 Preferred Connection Protocol Type

DNS uses Transmission Control Protocol (TCP) for zone transfers and User Datagram Protocol (UDP) for queries. The reason behind it is that UDP is a much faster protocol as it works stateless. With normal behaviour of DNS, it is expected from a recursive resolver to query an NS with high percentages of UDP packets unless the connection is not reliable and a stateful protocol is needed, or there is a zone transfer.

From this property of DNS queries, it can be beneficial to look into the preferred connection type of a resolver. If there is a class which oddly behaving resolvers according to this standard behaviour are concentrated, this feature can help to explain the properties of the class better. To be able to add it as a feature, similar to preferred name server feature, shares of UDP and TCP preference from each IP address was counted and divided into the total query number from each distinct IP address. Then, the biggest portion was selected and added to the preferred connection protocol type feature. Again this feature has percentage representations between 0.5 and 1.

### 3.4.12 Time to Live (TTL) Value Analysis from IP Packet Header

TTL or maximum hop count is a predefined value in IP packet headers which limits the maximum hops an IP packet can make. This value is used for preventing a packet from looping endlessly in a network if the destination is not found. Each time a packet reaches an intermediate network device such as a router, this value is decreased. Each operating system uses different TTL starting values when sending a packet. As discussed in a workshop on Data Mining for Computer Security in 2003, using TTL values is a valid method [57] on operating system fingerprinting. Of course, operating system fingerprinting is out of the scope of this research but, just by looking at TTL values, this already gives a rough estimation on the type of operating system. Also, from the domain knowledge, each sectors' choice on the operating system to run the recursive resolver on can differ related to a lot of different variables. This feature, therefore, can be useful to distinguish different classes related to different sectors. For this purpose, TTL values of Windows, Linux&Mac and FreeBSD was identified. This is done according to the means of each IP addresses' TTL values. While running this analysis, a range of TLL values will be assigned to the operating systems. The reason behind this is the fact that in each hop the packet is travelling before its final destination, the TTL value is decreased by one.

#### 3.4.12.1 GNU/Linux&MacOS Operating Systems According to TTL

The initial TTL value of GNU/Linux and MacOS operating systems are set to 64 by default [58]. According to this, all IP addresses using a mean TTL value of (30,64) on day March 20, 2019 has marked as GNU/Linux&MacOS operating systems and a feature named lin_mac_ttl was created and value of 1 was given. For all other IP addresses, the value of 0 was given to represent that they do not share this feature.

##### 3.4.12.2 FreeBSD Operating Systems According to TTL

The initial TTL value of the FreeBSD operating systems is set to 255 by default [58]. According to this, all IP addresses using a mean TTL value between [200,255] on day March 20, 2019 was marked as FreeBSD operating systems and a feature named freebsd_ttl was created, and value of 1 was given. For all other IP addresses, the value of 0 was given to represent that they do not share this feature.

##### 3.4.12.3 Windows Operating Systems According to TTL

The initial TTL value of Windows operating systems is set to 128 by default [58]. According to this, all IP addresses using a mean TTL value between [64,128) on day March 20, 2019 was marked as Windows operating systems and a feature named windows_ttl was created, and value of 1 was given. For all other IP addresses, the value of 0 was given to represent that they do not share this feature.

##### 3.4.12.4 Other Operating Systems According to TTL

The initial TTL values of other operating systems are selected to close the gap on remaining unlabelled means of TTL values which are seen in .nl ccTLD NSes. According to this, all IP addresses using a mean TTL value between [0,30] and [128,199] on day March 20, 2019 was marked as other_ttl, and a feature named other_ttl was created, and value of 1 was given. For all other IP addresses, the value of 0 was given to represent that they do not share this feature.

### 3.4.13 'www.' Usage in the Query

From a recent patent from the United States on distinguishing human-driven DNS queries from machine-to-machine DNS queries [59], it is suggested that if a query includes a "www." it is more likely that the query is coming from a human-driven source than a machine to machine communication.

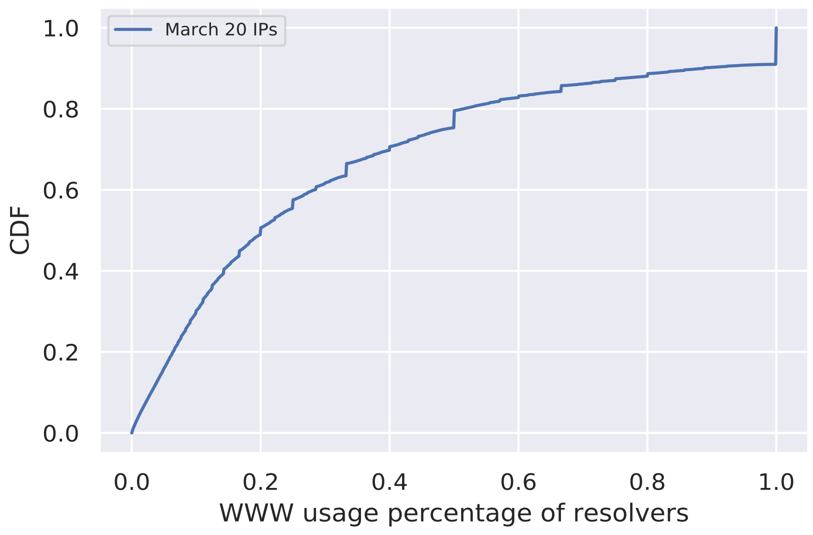


Figure 18: CDF Graph of Percentages of WWW Usages of IP Addresses on day March 20, 2019

From the Figure 18, it can be seen that if we select 80% of all IP addresses on day March 20, 2019, at least 50% of their queries includes “www.”. The results were encouraging to create a feature for the “www.” usage in the data set. To be able to do it counts of “www.” including queries are divided into the total number of queries on the aforementioned day for each distinct IP connected to NSs of .nl and added as a feature.

### 3.4.14 Data Set Creation Wrap-Up

Overview of all the created features are listed in the Table 2 below with their short explanations.

| **Feature Name** | **Explanation** |
|---|---|
| ***rcode0*** | Response Code: No Error |
| ***rcode3*** | Response Code: NXDomain (Non eXisting Domain) |
| ***qtype1*** | RR Type: A (IPv4Address) |
| ***qtype2*** | RR Type: NS (Name Server) |
| ***qtype5*** | RR Type: CNAME (Canonical Name) |
| ***qtype6*** | RR Type: SOA (Start of Authority) |
| ***qtype15*** | RR Type: MX (Mail eXchange) |
| ***qtype16*** | RR Type: TXT |
| ***qtype28*** | RR Type: AAAA (IPv6 Address) |
| ***qtype33*** | RR Type: SRV (Server Selection) |
| ***qtype43*** | RR Type: DS (delegation Signer) |
| ***qtype46*** | RR Type: RRSIG (Resource Record SIGnature) |
| ***qtype48*** | RR Type: DNSKEY |
| ***aa1*** | Authoritative Answer |
| ***rd1*** | Recursion Desired |

| | |
|---|---|
| ***qname_mini*** | Percentage of Qname Minimised Queries |
| ***domain_cover*** | Percentage of Daily Domain Cover |
| ***port_randomness*** | Port Randomness |
| ***edns_do*** | Boolean EDNS_DO flag |
| ***cd1*** | Checking Disabled |
| ***max_value_ns*** | Which Name Server is Queried Most? |
| ***max_value_prot*** | Which Protocol Has More Percentage? |
| ***lin_Mac_ttl*** | Queries from Lin or Mac Servers |
| ***windows_ttl*** | Queries from Windows Servers |
| ***freebsd_ttl*** | Queries From FreeBSD Devices |
| ***other_ttl*** | Unknown TTL Range |
| ***www_usage*** | Percentage of queries which include “www.” |

Table 2: Created Feature Set

When all the features of the feature set are created, the total number of 27 features were in the set. 17 of them were selected directly from ENTRADA database of DNS queries which includes the raw properties of DNS packets and 10 features were derived from the DNS packets’ information, meaning they were created by using one or more columns of the DNS queries database to be able to aggregate a different result than any other column in the database.

As the next step, these features will be analysed to see each feature’s contribution to the classification case. According to this analysis, it will be decided which features are going to be kept in the feature set for classification.

## 3.5 Feature Analysis & Elimination

Feature analysis is one of the “must-do”s of the researches with machine learning. Even though with the processing power of today’s computers there are not too many dimensions (features) in the data set, unnecessary features still consume resource and they might cause the machine learning algorithm to over-fit on the ground truth data set instances.

Therefore, two different feature analysis algorithms were used to reduce the dimensions of the data set. They will be discussed in the following subsections.

### 3.5.1 Univariate Feature Selection

As mentioned in the Related Work Section of 2.4.2, a chi-square test is one of the most famous tests in feature selection. This method is automatised in scikit-learn as univariate feature selection. Univariate feature selection works by selecting the best features based on univariate statistical tests [60].

| Ranking | Columns | Score |
|---|---|---|
| 1 | *windows_ttl* | 2257.309084 |
| 2 | *lin_Mac_ttl* | 1406.279070 |
| 3 | *other_ttl* | 780.221277 |
| 4 | *edns_do* | 584.637710 |
| 5 | *cd1* | 399.660342 |
| 6 | *qtype43* | 182.229372 |
| 7 | *qtype28* | 179.462047 |
| 8 | *qname_mini* | 117.982002 |
| 9 | *aa1* | 102.641688 |
| 10 | *www_usage* | 77.579091 |
| 11 | *port_randomness* | 65.672585 |
| 12 | *freebsd_ttl* | 65.088744 |
| 13 | *qtype15* | 33.806447 |
| 14 | *qtype6* | 31.467198 |
| 15 | *qtype2* | 31.258502 |
| 16 | *max_value_ns* | 29.989282 |
| 17 | *qtype1* | 29.630239 |
| 18 | *qtype16* | 24.407651 |
| 19 | *rd1* | 17.592141 |
| 20 | *qtype48* | 15.421381 |

Table 3: Result of feature importance with $\chi^2$ Test on data set created on day March 20, 2019

It is a pre-processing step for an estimator. Scikit-learn exposes feature selection routines as objects that implement the transform method. It is also mentioned that for classification purposes, one of chi-square, f_classif or mutual_info_classif tests should be used [61].

To analyse the importance of the features, the test was run in combination with the chi-squared test. All IP addresses from the ground truth were searched in the previously-created data set with 27 features, and matching rows are copied to another data set. Then along with each IP addresses' target classes, this new filtered dataset was fed to the algorithm. Results can be seen from Table 3.

Given dataset about two events, we can get the observed count O and the expected count E. Chi-square score measures how much the expected counts E and observed Count O derive from each other one by one for each feature. Therefore, it is a reliable statistical and deterministic test.

### 3.5.2 Tree-based Feature Selection

Extra trees classifier test is yet another method in feature elimination, which is under the class of Tree-based feature selection. The explanation of this method in scikit-learn's web page was: "Tree-based estimators can be used to compute feature importance, which in turn can be used to discard irrelevant features (when coupled with the sklearn.feature_selection.SelectFromModel" [62].

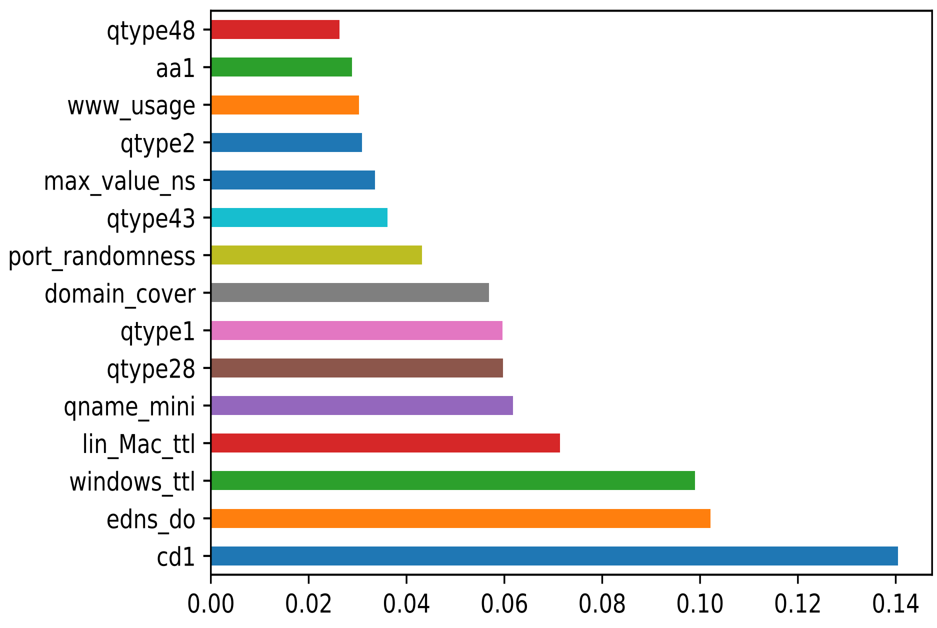


Figure 19: Result of feature importance with Extra Trees Classifier on data set created on day March 20, 2019

As suggested in the explanation, a combination of select from model is also included in the algorithm. Similar to the chi-squared test, all IP addresses from the ground truth was searched in the previously-created data set with 27 features, and matching rows are copied to another data set. Then along with each IP addresses' target classes, this new filtered dataset was fed to the algorithm. Results can be seen in Figure 19.

### 3.5.3 Selected Features

From the Table 3 and Figure 19, it can be seen that in the top 15 most important features list, following 11 features are common for both tree-based and univariate feature selection:

- windows_ttl: Windows TTL
- lin_Mac_ttl: Linux-Mac TTL
- edns_do: EDNS DO bit set to 1
- cd1: Checking Disabled bit set to 1
- qtype43: Shares of DS RR type queries
- qtype28: Shares of AAAA RR type queries
- qname_mini: Shares of qname minimisation compliant queries
- aa1: Authoritative Answer bit set to 1
- www_usage: Shares of “WWW.” usage in the query
- port_randomness: Shares of standard deviations of incoming port numbers

- qtype2: Shares of NS RR type queries

From analysing the results of two different importance analysis tests, the feature set was decided to be decreased to 15 features. Therefore for the rest four features that differ in both tests, a decision should be made on which approach is going to be used for feature elimination. It is decided to include features from chi-square test because of the following two main reasons:

- One of the most important (3rd in ranking) feature in chi-square test which is the other_ttl, is not included in the tree-based feature selection as in the tree-based approach the decision tree forming process is made randomly.
- Similarly, as the tree-based algorithm is not deterministic, each time the test was run, the least five important features from the top 15 features list were changing.

These two reasons encouraged the selection to incline to the part of deterministic approach. So the first 15 features from the Table 3 was selected as the final feature set for the classification.

## 3.6 Applied Machine Learning Algorithms with Python

By the time this research was being conducted, there were lots of different classification techniques which were already implemented in the scikit-learn library in Python. In the official web page of scikit-learn, there is a cheat-sheet which can be seen in the Figure 20 below.

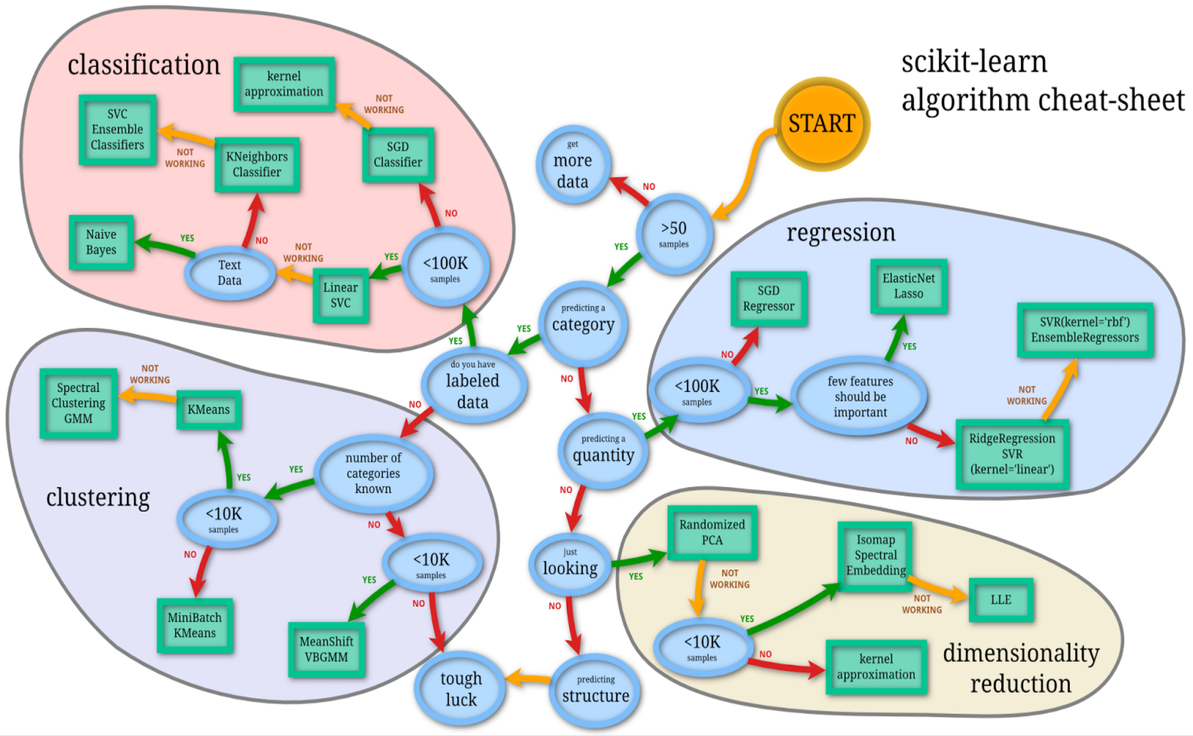


Figure 20: Algorithm selection cheat sheet from scikit-learn [5]

Even though cheat-sheet was a good point for a quick start for algorithm selection, some other popular classification algorithms were tried as well. The reason behind it is a survey by Guyen et al. named “Techniques for Internet Traffic Classification using Machine Learning” [23]. In the survey, all recent papers on traffic analysis using machine learning is summarised. In the survey, the research showed that k-nearest neighbours, neural networks and decision tree-based algorithms were the most popular algorithms. That is also why all of these algorithms will be tried on the prepared data set. In the subsections of this section, machine learning algorithms which were used in this research and their properties will be discussed.

### 3.6.1 Support Vector Machines (SVM)

In the Figure 20, if followed from the start for the case of this research, the first algorithm that appears for the classification is Linear Support Vector Classifier (SVC). So the classification case has started with this one. According to scikit-learn, advantages of SVM based classification methods are as follows,

- “Effective in high dimensional spaces.” even though the feature set is not too big, efficient algorithms can increase the accuracy of the classification.
- “Still effective in cases where the number of dimensions is greater than the number of samples.” which is not the case in this research
- “Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.” which is important and valid for any classification case.
- “Versatile: different Kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.” which is also useful as classification can be tested with different Kernel functions.

In SVMs there are three functions which can be used for multi-class classification: SVC, NuSVC and LinearSVC. SVC and NuSVC are similar methods, but accept a bit different sets of parameters and have different mathematical formulations. Therefore, NuSVC will be skipped and will not be applied to the dataset as SVC will already be used. On the other hand, LinearSVC is a different implementation of SVC, for the case of a linear kernel.

For multi-class classification, SVC implements the “one-against-one” approach, as Knerr et al. suggested. He performed pair-wise comparisons between all n classes. Therefore, from the n classes in the training set, all possible two-class classifiers are evaluated. In this method, each classifier is trained on only two-out-of-n classes, so it gives a total of $n(n-1)/2$ classifiers [63]. On the other hand, LinearSVC implements “one-against-the-rest” as a multi-class strategy. Hence $n$ class models are trained for each classification case. If there are only two target classes, only one model is trained.

### 3.6.2 k-Nearest Neighbours

Scikit-learn provides functionality for both supervised and unsupervised neighbour(s)-based learning methodologies. The main ideology of the nearest neighbour models is to find a (predefined) number of samples from the training set which are closest to the new point and thereby predict the label from these hints. For this purpose, before the algorithm is executed, a variable (k) which defines the number of neighbours should be passed to the algorithm.

The distance used in this algorithm can be nearly any distance calculating metric. The most common distance metric is Euclidean distance. Neighbour based algorithms are often called as non-generalising machine learning approaches as they save each point in the memory from training data for further use to find k-nearest neighbours of each point.

For finding the optimal k, mean errors can be calculated by dividing training set into train and test sets. Now as there will be train and test sets which are all formed from a training set, all sets are labelled set. So the algorithm can be taught on train and errors can be calculated from the test set, which is a subset of the training set in this case.

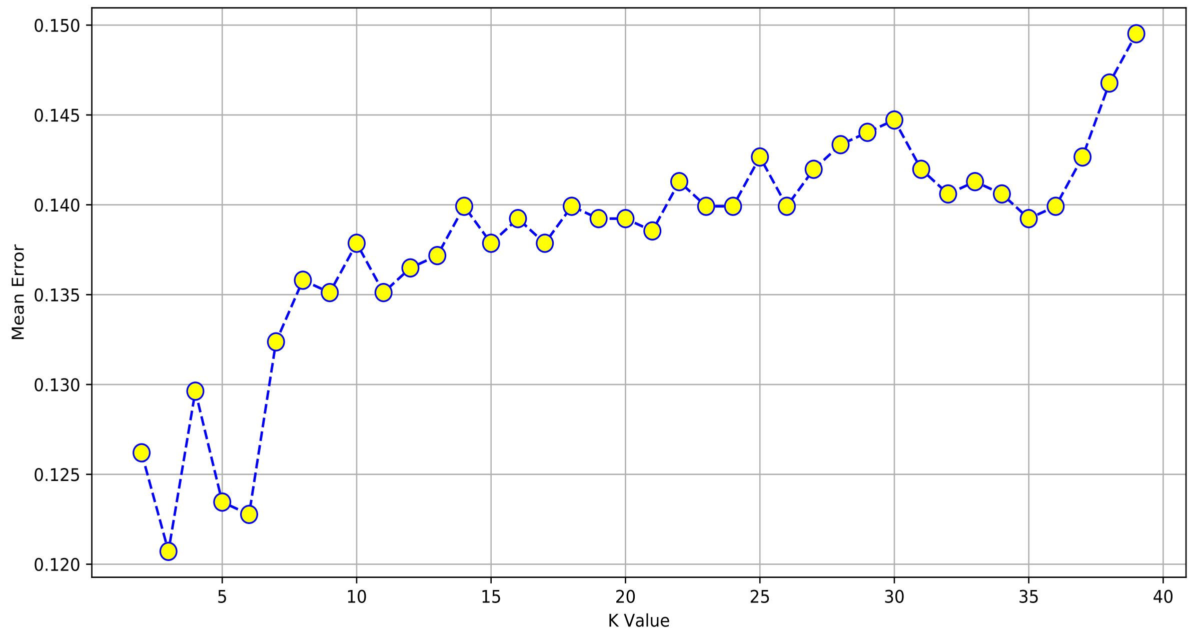


Figure 21: Error rate of K values

From the Figure 21, it can be clearly seen that starting from k=2, if k is chosen as 3, the minimum error rate is obtained. Thus, in the classification, k will be set to 3 to obtain the least error version for classification.

### 3.6.3 Neural Networks

Classification with neural networks in scikit-learn is done with Multi-layer Perceptron (MLP). The advantages of Multi-layer Perceptron are:

- Capability to learn non-linear models.
- Capability to learn models in real-time using partial_fit parameter

Class MLPClassifier in scikit-learn library implements an MLP algorithm that trains using back-propagation. The main feature of the back-propagation algorithm is its iterative, recursive and efficient method for calculating the weights updates. It improves the network until it is able to perform the task which it is being trained for [64].

MLPClassifier trains on two arrays. The first array is (n_samples, m_features), which includes the training data, and the second array is (n_samples), which holds the class labels for the training instances. In the used algorithm, {*solver='sgd', hidden_layer_sizes=(20,15,10)*} was chosen as parameters. The solver is used for weight optimisation, and 'sgd' refers to stochastic gradient descent. Hidden layer sizes determine the number and size of hidden layers. In this approach, three hidden layers size of 20,15 and 10 were used respectively for hidden layers number 1,2 and 3.

### 3.6.4 Random Forest Classifier

Random Forest classifier is a part of the ensemble module of scikit-learn. This algorithm is specifically designed for trees meaning various classifier sets are created because of the randomness embedded in the classifier structure. The prediction of the ensemble is given as the averaged prediction of the individual classifiers. Likely to most of the other classifiers, random forest classifier is also fitted with two arrays: an array X of size [n_samples, m_features] which holds the training samples, and an array Y of size [n_samples] holding the class labels for the provided training set [25].

In random forest, while the building the tree, when splitting a node into pieces, the split that is chosen is not the best split among all features. Instead of that, the split which is chosen is the best split among a random subset of the features. This randomness generally leads an increase in the bias of the forest (with respect to the bias of a single non-random tree). On the other hand, as there is averaging, the variance decreases more than enough for compensating for the increase in bias, so in the end, creates a better model than a single non-random tree [25].

For the efficiency purposes, from the model selection class of scikit-learn, GridSearchCV function is used to be able to find the best hyper_parameters of the random forest for the dataset being used for conducting this research. Two different methods are used to test parameters:

- (3x4)=12 combinations of hyper-parameters:
  - 'n_estimators': [3, 10, 30]
  - 'max_features': [2, 5, 6, 8]
- (3x2)=6 combinations of hyper-parameters with bootstrap set to 'false':
  - 'n_estimators': [3, 10]
  - 'max_features': [2, 3, 4]

Highest accuracy for best parameters came from the twelve combination part where: 'max_features' were 5, and 'n_estimators' were 30. In the classification case, these parameters are going to be fed to the function to get the best accuracy.

# 4 Results

In this section, results of machine learning algorithms are going to be discussed. Accuracies and related statistics will be shared and answers to second and third research questions are going to be provided.

## 4.1 Labelled Data Set Results on Different Machine Learning Algorithms

The statistics of machine learning algorithms are shared in the subsections of this section. The IP addresses which are used in this part are obtained by searching IP addresses of the ground truth data on the created data set on day March 20, 2019. From a total number of 39,361 labelled ground truth IP addresses 7,287 of them were in the data set created on the day as mentioned earlier. So, all IP addresses found are removed with all of its features to create a smaller labelled data set. This new set divided into a training set (80%, 5829 distinct IP addresses) to train the classifier on and test set (20%, 1458 distinct IP addresses) to use the classifier for prediction so that statistics can be seen in the real data. In the shared confusion matrices, the x-axis represents the predicted classes and y-axis represents correct classes.

### 4.1.1 Support Vector Machines (SVM)

In Section 3.6.1, it was mentioned that in SVM, there are two algorithms which follow different approaches for the classification from support vectors. The first one to be analysed is Linear SVM.

| True/Predicted | Cloud | Hosting | ISP | IT Enterprise | Research | Telecommunication | Open Resolvers |
|---|---|---|---|---|---|---|---|
| Cloud | 5 | 0 | 32 | 6 | 0 | 0 | 4 |
| Hosting | 0 | 0 | 23 | 10 | 0 | 0 | 3 |
| ISP | 0 | 0 | 300 | 43 | 0 | 0 | 24 |
| IT Enterprise | 0 | 0 | 25 | 172 | 0 | 0 | 1 |
| Research | 0 | 0 | 10 | 10 | 0 | 0 | 3 |
| Telecommunication | 0 | 0 | 2 | 4 | 0 | 0 | 1 |
| Open Resolvers | 0 | 0 | 61 | 4 | 0 | 0 | 715 |

Table 4: Linear SVC Classifier confusion matrix on the ground truth data set

From the confusion matrix in Table 4, it can be seen that the confusion matrix returned from Linear SVM has one of the most misclassified class instances. The green cells show the correct classification for each class type. In a faultless classification case, all the data is expected to be in the green cells. Linear SVC has completely failed to classify any instances from Hosting, Research and Telecommunication classes. Therefore all instances which belong to those classes are misclassified.

For this classifier, the most confused classes are “ISP – IT Enterprise” and “ISP – Open Resolvers”. The confusion in classes ISP and open resolver is the main confusion for almost all the classifiers. The reason behind this can be the fact that both classes are serving hundreds of thousands of end-users so there is a big chance that they might follow same structures at some points as they are all considered to run well-maintained resolving services. In addition to this, classifier seems to be biasing its wrong predictions to the favour of ISP. This can be observed by looking at the x-value of the ISP section. Most of the misclassified values are predicted to be ISP resolvers.

| | Precision | Recall | F-1 Score | Support |
|---|---|---|---|---|
| Cloud | 1 | 0.11 | 0.19 | 47 |
| Hosting | 0 | 0 | 0 | 36 |
| ISP | 0.66 | 0.82 | 0.73 | 367 |
| IT Enterprise | 0.69 | 0.87 | 0.77 | 198 |
| Research | 0 | 0 | 0 | 23 |
| Telecommunication | 0 | 0 | 0 | 7 |
| Open Resolvers | 0.95 | 0.92 | 0.93 | 780 |
| Weighted Average | 0.80 | 0.82 | 0.80 | 1458 |

Table 5: Linear SVC classifier classification report on the ground truth data set

From the classification report in Table 5, it can also be seen clearly that for Hosting, Research and Telecommunication classes precision, recall and F-1 scores are zero. The algorithm's bias for predicting the misclassified instances as ISP resolvers shows impact in the classification report by looking at the difference between precision and recall values. The reason behind this is, precision is a value calculated by dividing the correctly classified data from a class into all instances which are classified as that class. If there is a big difference between those numbers, it always affects the precision to result in lower percentages.

The second approach in classification with SVM is SVC classifier, which follows one-against-one approach. This approach also does not seem like much better than linear SVC. Following two tables are showing the results from this classifier.

| True/Predicted | Cloud | Hosting | ISP | IT Enterprise | Research | Telecommunication | Open Resolvers |
|---|---|---|---|---|---|---|---|
| Cloud | 5 | 0 | 31 | 6 | 1 | 0 | 4 |
| Hosting | 0 | 0 | 25 | 10 | 1 | 0 | 0 |
| ISP | 0 | 0 | 306 | 40 | 0 | 0 | 21 |
| IT Enterprise | 0 | 0 | 19 | 178 | 0 | 0 | 1 |
| Research | 0 | 0 | 10 | 10 | 1 | 0 | 2 |
| Telecommunication | 0 | 0 | 2 | 4 | 0 | 0 | 1 |
| Open Resolvers | 0 | 0 | 54 | 2 | 0 | 0 | 724 |

Table 6: SVC Classifier confusion matrix on the ground truth data set

From the Table 6, it can be seen that the bias in favour of the ISP class continues in SVC classifier. However, in combination with Table 7, it can be interpreted that this bias slightly decreased as precision and recall values are slightly increased. On the other hand, now there is one correct classification in the research class, which immediately showed the effects on classification report table. Overall, the precision of the algorithm is 81%

As a short conclusion for SVM based learning algorithms, it is not the right choice for resolver classification case. One reason for this algorithm not being suitable for resolver classification might be coming from the nature of the algorithm that SVMs are not primarily designed for multi-class classification cases.

| | Precision | Recall | F-1 Score | Support |
|---|---|---|---|---|
| Cloud | 1 | 0.11 | 0.19 | 47 |
| Hosting | 0 | 0 | 0 | 36 |
| ISP | 0.68 | 0.83 | 0.75 | 367 |
| IT Enterprise | 0.71 | 0.90 | 0.79 | 198 |
| Research | 0.33 | 0.04 | 0.08 | 23 |
| Telecommunication | 0 | 0 | 0 | 7 |
| Open Resolvers | 0.96 | 0.93 | 0.94 | 780 |
| Weighted Average | 0.82 | 0.83 | 0.81 | 1458 |

Table 7: SVC classifier classification report on the ground truth data set

### 4.1.2 k-Nearest Neighbours

K-Nearest Neighbour algorithm was the second suggested algorithm in case the SVM algorithm does not work for the data set researcher is working on. Therefore, this algorithm was the second one that applied to the dataset.

| True/Predicted | Cloud | Hosting | ISP | IT Enterprise | Research | Telecommunication | Open Resolvers |
|---|---|---|---|---|---|---|---|
| Cloud | 29 | 4 | 10 | 1 | 1 | 0 | 2 |
| Hosting | 8 | 18 | 8 | 1 | 1 | 0 | 0 |
| ISP | 25 | 19 | 308 | 7 | 1 | 0 | 7 |
| IT Enterprise | 5 | 5 | 15 | 173 | 0 | 0 | 0 |
| Research | 3 | 3 | 11 | 3 | 3 | 0 | 0 |
| Telecommunication | 1 | 0 | 3 | 0 | 1 | 2 | 0 |
| Open Resolvers | 0 | 2 | 27 | 2 | 0 | 0 | 749 |

Table 8: k-Nearest Neighbours Classifier confusion matrix on the ground truth data set

Just by looking at the confusion matrix in Table 8, it can be seen that the misclassified instances are decreased significantly with the k-nearest neighbour algorithm. With this algorithm, the situation of the misclassified instances' bias which was mostly in favour of ISP class has changed, and misclassification bias is now distributed to Cloud and Hosting classes as well, significantly improving the precision of ISP class.

From the confusion matrix, it can also be seen that the classifier can now find resolvers for each class, and there is no class that there are no correct predictions.

|  | Precision | Recall | F-1 Score | Support |
|---|---|---|---|---|
| Cloud | 0.41 | 0.62 | 0.49 | 47 |
| Hosting | 0.35 | 0.50 | 0.41 | 36 |
| ISP | 0.81 | 0.84 | 0.82 | 367 |
| IT Enterprise | 0.93 | 0.87 | 0.90 | 198 |
| Research | 0.43 | 0.13 | 0.20 | 23 |
| Telecommunication | 1 | 0.29 | 0.44 | 7 |
| Open Resolvers | 0.99 | 0.96 | 0.97 | 780 |
| Weighted Average | 0.88 | 0.87 | 0.87 | 1458 |

Table 9: k-Nearest Neighbour classifier classification report on the ground truth data set

By looking at the classification report in Table 9, it can be seen that the significant positive changes in the correct predictions are also affected the scores in the table in each class and in the end, overall. The lowest score in correctly classified classes belongs to "Research" class where just 3 out of 23 resolvers are classified correctly. This is most probably happening because there is not enough number of "research class" examples to train on. Highest accuracy in the classification belongs to open resolvers class with 99% of precision and 96% of recall value. It can be seen that Open Resolvers class is the class that which forms the biggest portion and the most precise labelling in the ground truth data.

### 4.1.3 Neural Networks

Neural networks are efficient in learning as it distributes the errors back to its weight vectors to decrease the error rate and increase the accuracy of the classifier. The results of Neural Network-based classification can be seen in the following two Tables.

| True/Predicted | Cloud | Hosting | ISP | IT Enterprise | Research | Telecommunication | Open Resolvers |
|---|---|---|---|---|---|---|---|
| Cloud | 5 | 0 | 35 | 3 | 0 | 0 | 4 |
| Hosting | 0 | 0 | 26 | 8 | 1 | 0 | 1 |
| ISP | 0 | 0 | 320 | 30 | 0 | 0 | 17 |
| IT Enterprise | 0 | 0 | 33 | 164 | 0 | 0 | 1 |
| Research | 0 | 0 | 12 | 9 | 1 | 0 | 1 |
| Telecommunication | 0 | 0 | 3 | 3 | 0 | 0 | 1 |
| Open Resolvers | 0 | 0 | 70 | 5 | 0 | 0 | 705 |

Table 10: Neural Networks Classifier confusion matrix on the ground truth data set

From the Table 10, it can be seen that the neural networks algorithm is also biased to classify wrong predictions as ISP resolvers. There are no instances classified as Hosting or Telecommunication, which makes the classifier as ineffective as SVM based classification algorithms mentioned in the previous subsections.

|  | Precision | Recall | F-1 Score | Support |
|---|---|---|---|---|
| Cloud | 1 | 0.11 | 0.19 | 47 |
| Hosting | 0 | 0 | 0 | 36 |
| ISP | 0.64 | 0.87 | 0.74 | 367 |
| IT Enterprise | 0.74 | 0.83 | 0.78 | 198 |
| Research | 0.50 | 0.04 | 0.08 | 23 |
| Telecommunication | 0 | 0 | 0 | 7 |
| Open Resolvers | 0.97 | 0.90 | 0.93 | 780 |
| Weighted Average | 0.82 | 0.82 | 0.80 | 1458 |

Table 11: Neural Networks classifier classification report on the ground truth data set

If looked at the Table 11, one can see that the difference between precision and recall which represents the difference between the share of correctly classified data from all the predictions of that class versus shares of correctly classified data from that class among all members of the class in the dataset. In a classification case, it is best if the difference between precision and recall is low, but the values of them are high.

### 4.1.4 Random Forest Classifier

Random forest classifier is one of the most common classification technique because of the reasons explained in Section 3.6.4. Among five different algorithms that were applied to the ground truth data set, random forest classifier is the best in terms of accuracy.

| True/Predicted | Cloud | Hosting | ISP | IT Enterprise | Research | Telecommunication | Open Resolvers |
|---|---|---|---|---|---|---|---|
| Cloud | 25 | 1 | 16 | 1 | 2 | 0 | 2 |
| Hosting | 1 | 16 | 16 | 1 | 2 | 0 | 0 |
| ISP | 5 | 4 | 333 | 7 | 2 | 0 | 16 |
| IT Enterprise | 0 | 1 | 16 | 180 | 0 | 0 | 1 |
| Research | 0 | 3 | 6 | 1 | 11 | 0 | 2 |
| Telecommunication | 0 | 0 | 2 | 0 | 2 | 1 | 1 |
| Open Resolvers | 0 | 0 | 11 | 1 | 0 | 0 | 768 |

Table 12: Random Forest Classifier confusion matrix on the ground truth data set

By looking at the confusion matrix in Table 12, it can be seen that there is just one instance classified as telecommunication. Even though this classification was correct, the other six instances are classified wrongly, which reduces the recall below 15%. On the other hand among all the other algorithms, the random forest is the classifier which had the most precise Research class classification, increasing the harmonic average (F-1 Score) of precision and recall above 50%.

There are also confusions in random forest classifier, especially between "Cloud & ISP". Cloud instances which are wrongly classified as belonging to ISP class forms just a bit less than the half the all instances which are labelled as Cloud in the ground truth. The condition is even worse for Hosting class. More than half of it was misclassified, and yet again, the bias is on the ISP class. The same number of correctly classified Hosting instances are classified as ISP resolvers.

| | Precision | Recall | F-1 Score | Support |
|---|---|---|---|---|
| Cloud | 0.81 | 0.53 | 0.64 | 47 |
| Hosting | 0.64 | 0.44 | 0.52 | 36 |
| ISP | 0.83 | 0.91 | 0.87 | 367 |
| IT Enterprise | 0.94 | 0.91 | 0.93 | 198 |
| Research | 0.58 | 0.48 | 0.52 | 23 |
| Telecommunication | 1 | 0.14 | 0.25 | 7 |
| Open Resolvers | 0.97 | 0.98 | 0.98 | 780 |
| Weighted Average | 0.91 | 0.91 | 0.91 | 1458 |

Table 13: Random Forest classifier classification report on the ground truth data set

Among all the other algorithms, the random forest has the highest F-1 scores for all class types, which can be seen from Table 13. Also, from the weighted averages of random forest classifier, it can be interpreted that only 9 resolvers are misclassified out of 100 instances, which makes the algorithm satisfactory for analysis on unlabelled data.

### 4.1.5 Algorithm Selection for Unlabelled Data

In the Table 14 below, F-1 scores of each classifier on each class type and weighted averages (total) is shared. The green cells are representing the highest score for each class type.

| | *F-1 Scores* | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Algorithm** | *Cloud* | *Hosting* | *ISP* | *IT Enterprise* | *Research* | *Telecommunication* | *Open Resolvers* | *Total* |
| **Linear SVC** | 0.19 | 0 | 0.73 | 0.77 | 0 | 0 | 0.93 | 0.80 |
| **SVC** | 0.19 | 0 | 0.75 | 0.79 | 0.08 | 0 | 0.94 | 0.81 |
| **k-Nearest Neighbours** | 0.49 | 0.41 | 0.82 | 0.90 | 0.20 | 0.44 | 0.97 | 0.87 |
| **Neural Networks** | 0.19 | 0 | 0.74 | 0.78 | 0.08 | 0 | 0.93 | 0.80 |
| **Random Forest** | 0.64 | 0.52 | 0.87 | 0.93 | 0.52 | 0.25 | 0.98 | 0.91 |

Table 14: F-1 Scores of each classifier on each class type.

There is no doubt that random forest is the best algorithm in terms of accuracy so far by having the best score for each class. Therefore, random forest classifier is selected as prediction algorithm on unlabelled data to make sure that the predictions have the best accuracy experimented.

Before using random forest on unlabelled data, further analysis is made on the predicted classes to analyse the prediction confidence of the classifier further. In random forest classifier class of sklearn.ensemble module, there is a method called predict_proba. For each instance in the test set which predictions are made on, it shows the membership values of each instance to predefined target classes. The head of the data can be seen in Table 15.

| | Cloud | Hosting | ISP | IT Enterprise | Research | Telecommunication | Open Resolvers |
|---|---|---|---|---|---|---|---|
| 1st Resolver in Data Set | 0.025 | 0.000 | 0.775 | 0.000 | 0.075 | 0.000 | 0.125 |
| 2nd Resolver in Data Set | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 1.000 |
| 3rd Resolver in Data Set | 0.000 | 0.425 | 0.225 | 0.050 | 0.275 | 0.025 | 0.000 |
| 4th Resolver in Data Set | 0.025 | 0.175 | 0.275 | 0.325 | 0.200 | 0.000 | 0.000 |

Table 15: Membership values of instances from the test set to the predefined classes.

Assume that an instance has equal distances to all target classes in the dataset, then, as there are 7 different target classes, the values of that instance in the predict_proba would be $1/7 = 0.142857143$. Therefore, any value that is higher than that value can be chosen as a predicted class. Having that in mind, the analysis of each instance from Table 15 is as follows:

- For the first resolver, it is predicted to be an ISP resolver, and the confidence on this prediction is 77.5%
- For the second resolver, it is predicted to belong to Open Resolvers class, and the confidence on this is 100%
- For the third resolver, it is predicted to be a Hosting resolver by 42.5% probability (confidence) as it has the highest probability among all the other classes. However, then, we can also say that the classifier is not that confident about this decision.
- For the fourth resolver the things get even fuzzier, it is predicted as an IT Enterprise resolver, but just with 32.5% of confidence. This resolver might also be an ISP or Research resolver.

To extend the research on this, for each predicted class means of confidence is calculated to be able to see the confidence of the classifier on each class. If the classifier is pretty confident in each class with high means, it means that this classifier is not even aware that it might be making errors. Otherwise, it means that classifier needs more training data to increase the confidence in the classes with low means.

| | Means of Confidence (Membership) Values for Each Class |
|---|---|
| Cloud | 0.7135 |
| Hosting | 0.5161 |
| ISP | 0.8227 |
| IT Enterprise | 0.8986 |
| Research | 0.4697 |
| Telecommunication | 0.4500 |
| Open Resolvers | 0.9660 |

Table 16: Mean of membership values of instances from the test set when classified to belong to each class.

From the Table 16, it is clear that most problematic classes in prediction which are Telecommunication and Research classes, having the lowest values of confidence during prediction. This is proof that more data is needed to increase the confidence of the classifier on those classes. On the other hand, the rest of the classes are predicted with much higher confidences except hosting. Even though hosting has a mean of more than 50%, among the rest four classes, it has the lowest mean than any other class.

### 4.1.6 Machine Learning Wrap-Up

Five different supervised machine learning algorithms are tried on the ground truth data set. There are some conclusions from the analysis of them. They can be found as follows:

- Nearly all algorithms are biased to classify an instance as ISP if not the real class.
- There is not enough data on Research and Telecommunication classes which result in really low accuracies compared to other classes. This should be improved for better results. On the other hand, this problem is also occurring because of the nature of those two classes, meaning there are not many autonomous systems in the wild that can be classified as those classes when compared to any other class in the target set.
- Cloud and Hosting classes need more data to train on to improve the recall values even though it is not critic to do so.
- Collecting precise ground truth data is time-consuming. Due to timing restrictions of the research, time spent on the ground truth formation was limited as well. Thus, some target classes did not have enough data to train on, which resulted in lower confidence levels in classification. Therefore, there were misclassified resolvers.
- Open Resolver instances are classified with a 97% of confidence, and it can be said that ground truth for this class is in the desired amounts.

The second research question which was “how to profile the recursive resolvers at an authoritative name server?” is now answered with the combination of feature creation & selection processes from Methodology section and showing results & interpreting different algorithm’s accuracies in the Results section.

As random forest provided the best scores among all categories, it will be used for classifying the unlabelled data to answer the third research question on “what are the main recursive resolvers of .nl NSes”.

## 4.2 Results on Unlabelled Data

### 4.2.1 Results on Day March 20, 2019

The third research question was asked to be able to run an experiment with the suggested methodology on a real-world example. Thus, the unlabelled data from March 20, 2019 which consist of 1,390,848 IP addresses are fed into the classifier after training on the whole ground-truth this time.

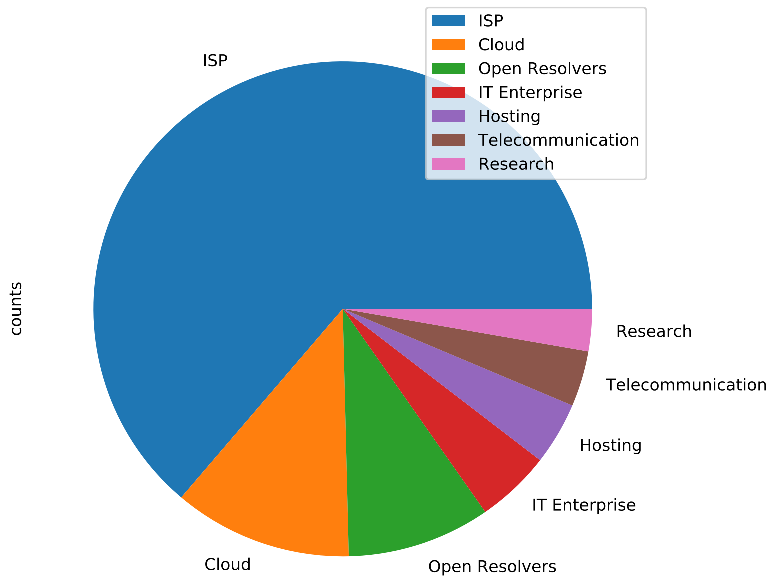


Figure 22: Recursive Resolver Distribution on March 20, 2019

According to the Figure 22, ISP resolvers are the main resolvers who contacts the .nl ccTLD NSes on day March 20, 2019. This forms 63.7% of all recursive resolvers. This is followed by Cloud resolvers with 11.6%. Then Open Resolvers and IT Enterprises form 9,2% and 4.8% respectively. The remaining 10% is shared between Hosting (4%), Telecommunication (3.6%) and Research (2.7%)

| | *Number of Instances* | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Confidence Interval Percentages** | *Cloud* | *Hosting* | *ISP* | *IT Enterprise* | *Research* | *Telecommunication* | *Open Resolvers* | *Total* |
| **[90,100]** | 220 | 2 | 57169 | 5253 | 7 | 0 | 14865 | 77516 |
| **[80,90)** | 2202 | 419 | 141606 | 4441 | 15 | 14 | 10974 | 159671 |
| **[70,80)** | 11891 | 1341 | 95558 | 6097 | 204 | 2224 | 15302 | 132617 |
| **[60,70)** | 33545 | 6120 | 141706 | 17877 | 1877 | 5055 | 42681 | 248861 |
| **[0,60)** | 110980 | 42874 | 447732 | 52308 | 30911 | 40870 | 46508 | 772183 |
| *Total:* | 158838 | 50756 | 883771 | 85976 | 33014 | 48163 | 130330 | 1390848 |

Table 17: Total number of IP addresses for each confidence interval on March 20, 2019

In the Table 17, the total number of IP addresses for each confidence interval on each target class is shared. According to the results in the table, for each class type, while confidence intervals are decreasing, the total number of IP addresses seen in the confidence interval is increasing. It can also be seen from the table that not all classes are having high numbers of IP addresses for where confidence percentage interval is higher than 90%.

In fact, the classifier labels instances from open resolvers with quite high confidence. 11.5% of all the instances classified as originating from open resolvers are having a confidence level of 90% or more. Other classes which have 6.5% and 6.1% of their total labelled IP addresses with confidence more than 90% are ISPs and IT Enterprises respectively. These results on the unlabelled data set are also following the same pattern as the Table 16 shared in the previous subsection.

On the other hand, there is no instance which Telecommunication resolvers are labelled with more than 90% of confidence. Other classes with low instances in more than 90% of confidence are Hosting, Research and Cloud classes with 0.004%, 0.02% and 0.2% respectively.

This confidence table, again, puts forth the fact that there is a need for more training instances on classes which have low values in high confidence intervals.

### 4.2.2 Class Patterns Analysis on Day March 20, 2019

In addition to other analysis, a box plot analysis is also run on a subset of features on predicted classes to be able to see the pattern differences among different class types. The methodology that is followed to draw the box plots can be seen in Figure 23.

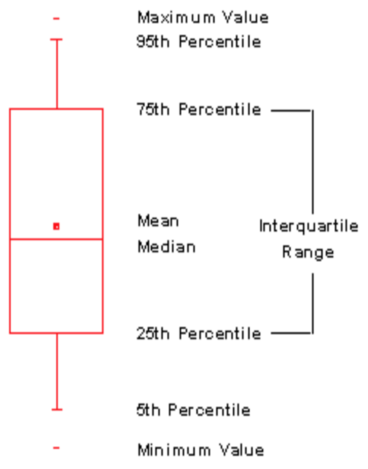


Figure 23: Box plot method [6]

In the figure, the lines that end with labels 5th Percentile and 95th Percentile is also called lower whisker and upper whisker, respectively. They represent 5% and 95% cuts of the data. Any point which has a value lower than 5th percentile or higher than 95th percentile is called outliers. Interquartile range (IQR) represents the distribution of middle 50% of the data. If there is no IQR in the plot, it means middle 50% is concentrated to the minimum or maximum levels. This can be interpreted according to the graph.

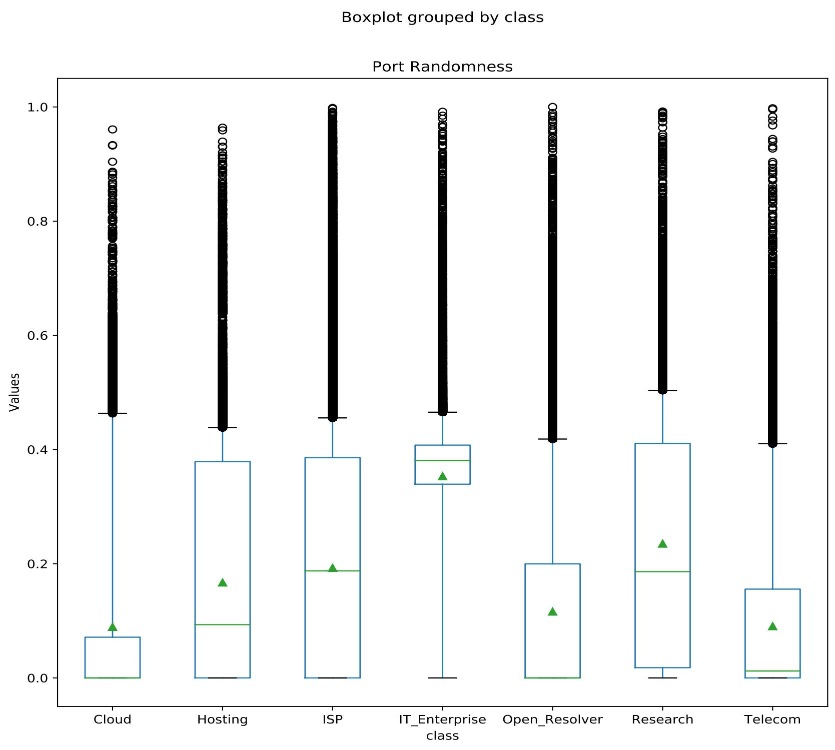


Figure 24: Box plot of port randomness feature amongst predicted classes on March 20, 2019

The first box plot on port randomness can be seen in Figure 24. All class types have outliers above the upper whisker. This shows that the remaining 5% of the data has a variety of values for each class type. Also, from the similarity on box plots of Hosting, ISP and Research classes, one of the reasons on confusion between those classes can be explained with this plot. This confusion in the class types can also be seen in Table 12 for a more explicit observation.

The second box plot, which is in Figure 25, is about query name minimisation on different class types. From the box plot, it is clear that most of the resolvers are having low values on query name minimisation. However, they are still showing different orientations on the shares of qname minimisation compliant queries. Telecommunication class, on the other hand, is having the most distinct behaviour here. This is because mean, median and 75th percentile lines all have a value of 0, which is the minimum value in the data. Therefore, there is also no IQR for Telecommunication. Furthermore, cloud and hosting resolvers are using qname minimisation in their queries more than any other class, which is making these classes distinctive for the classification case for this feature.

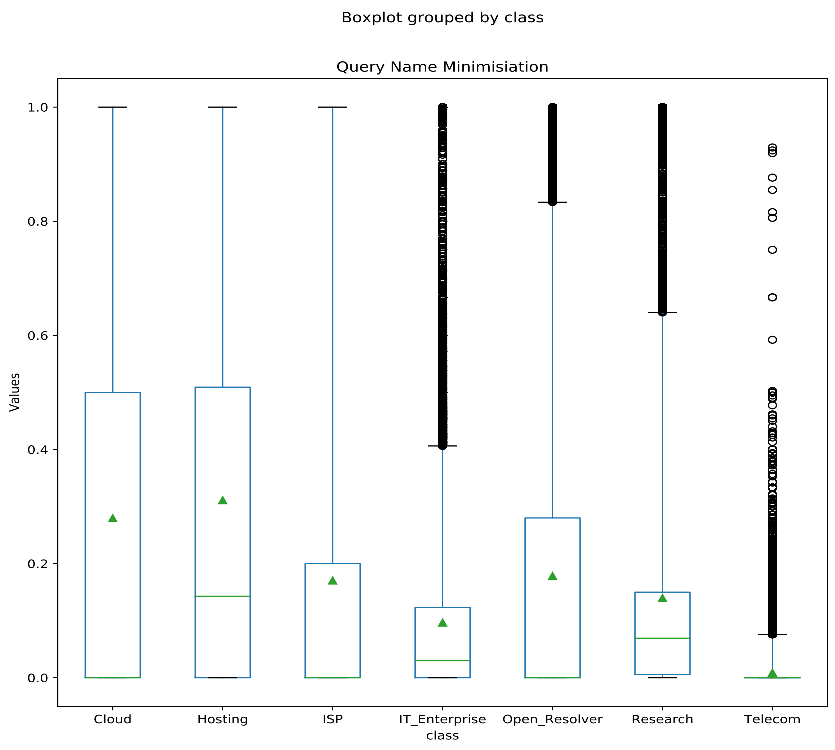


Figure 25: Box plot of qname minimisation feature amongst predicted classes on March 20, 2019

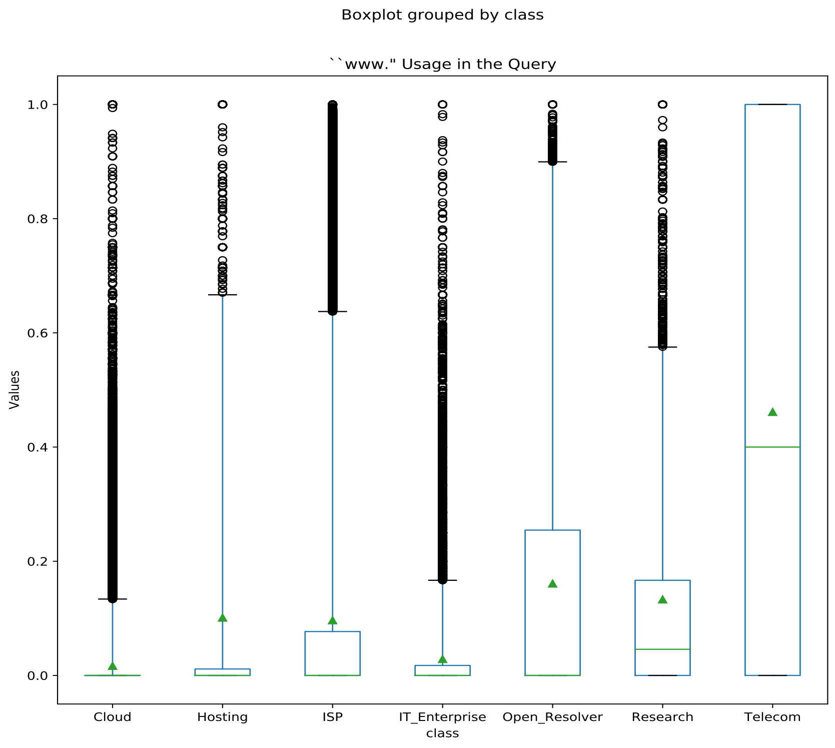


Figure 26: Box plot of “www.” usage feature amongst predicted classes on March 20, 2019

Another box plot on March 20, 2019 is about “www.” usage on each predicted class type. The most distinctive class of this feature is the telecommunication. All telecommunication related resolvers are having a wide variety of “www” usage in their queries. This also can be validated by looking at the mean of the plot, which is approximately at 0.5 level. Thus, this feature is distinctive for telecommunication class.

On the other hand, cloud and hosting related resolvers are not using “www.” frequently in their queries. This result is expected as cloud and hosting related resolvers are the ones which their shares of qname minimisation are rather high, and in DNS queries there is no need of using “www.” according to RFC 7816 [20]. It was also expected to see a high range of values on “www.” usage in open resolvers and ISP classes as they are mainly serving to end-users directly.

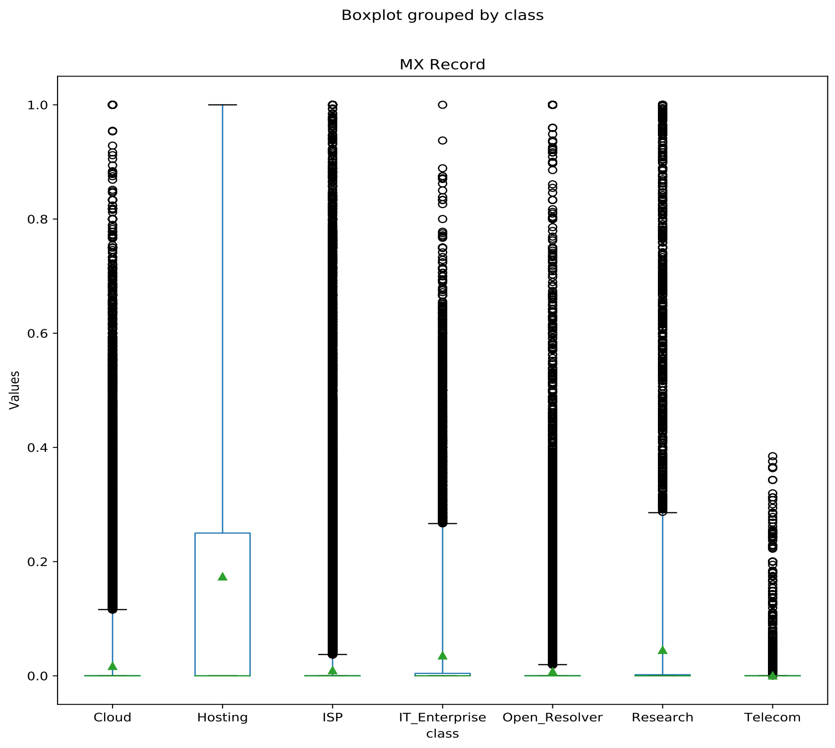


Figure 27: Box plot of MX record usage feature amongst predicted classes on March 20, 2019

The impact of MX usage on target classes was discussed in Section 3.4.2.6. It can be seen from the Figure 27 that, this feature is highly distinctive for Hosting class as discussed previously. Hosting has the highest mean among all classes on MX records shares with nearly 20%. In addition, IT Firms are the second most popular class querying for MX records with a mean of almost 6%.

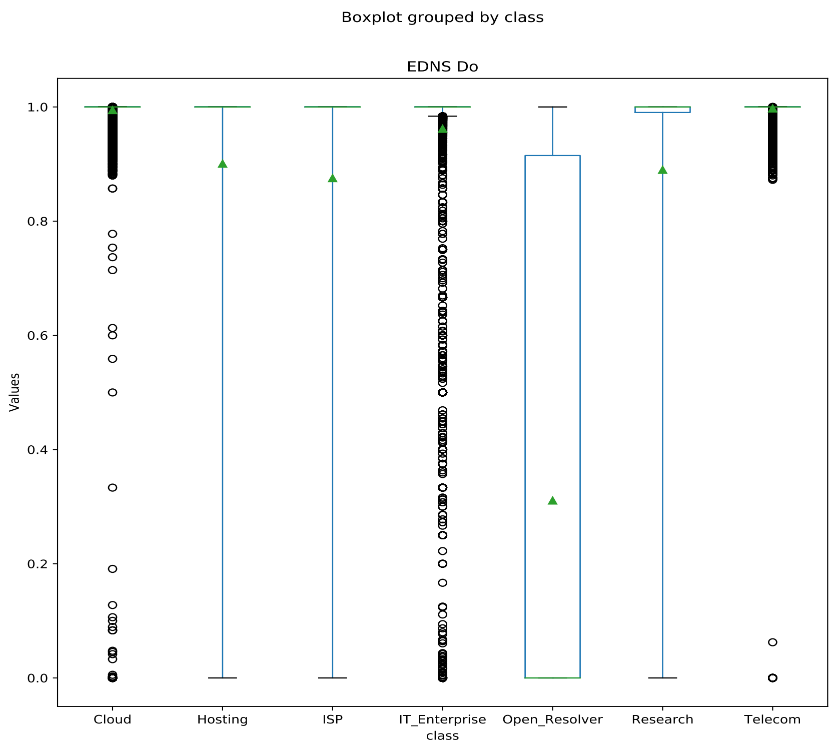


Figure 28: Box plot of EDNS-DO feature amongst predicted classes on March 20, 2019

Setting DNSSEC bit in a DNS query shows that recursive resolver is willing to do DNSSEC validation. From the Figure 28, box plot of EDNS-DO distribution amongst target classes can be seen. According to the plot, it is clear that all the target classes' resolvers are willing to perform DNSSEC validation process. However, in Open Resolvers class, there are lots of resolvers that do not set EDNS-DO bits in most of their queries. One reason for this can be the fact that there are lots of Open Resolvers on the internet that are available to the public because of some misconfigurations. As most of the aforementioned type of open resolvers are also not maintained by professionals, they are less likely to correctly apply security layers or significant operational changes in the DNS.

On the other hand, telecom and cloud classes' resolvers are willing to perform DNSSEC validation more than any other class type in the dataset. This behaviour can be observed by checking the mean and median values, which are approximately 100% for each class. In addition, most of the outlier resolvers originating from these classes are performing DNSSEC validation in more than 95% of their queries.

### 4.2.3 Results on Day May 22, 2019

To be able to collect the results, a new data set is created consisting of the same features as the dataset created on March 20, 2019. After creating the data set, all the IP addresses that are coming from an address in the ground truth data set are removed from this new set.

The unlabelled data from May 22, 2019 consisted of 1,366,634 IP addresses. The difference between the number of unlabelled unique resolvers on March 20, 2019 and on May 22, 2019 is 23854. This number does not form a percentage of more than 1.5% in either data sets. Therefore, a direct comparison of the number of IPs in each class should not lead to misinterpretations.

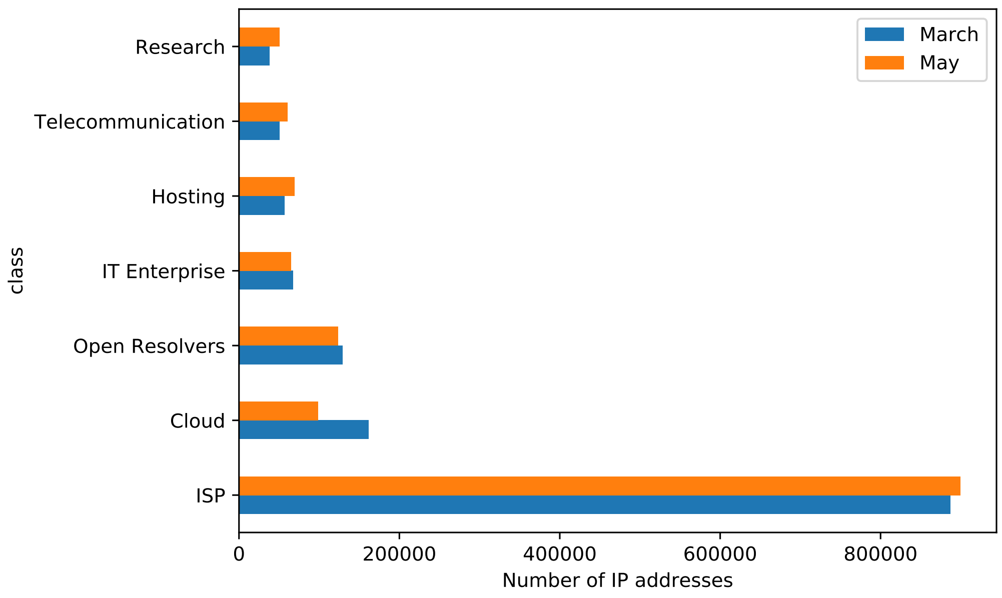


Figure 29: Number of IP addresses in each class from March 20, 2019 and May 22, 2019

To be able to classify the unlabelled data set from May, the new set is fed into the random forest classifier which has already trained on the ground truth instances on March 20, 2019. In the Figure 29 the number of IP addresses in each class from the predictions both in March and May are shared. In the past two months, the percentages of ISP resolvers had a noticeable increase from 63.7% to 65.8%. Most of this increase in ISP resolvers are originating from Cloud resolvers. Cloud resolvers had a 4.4% drop from March to May and dropped its share to 7.2%.

Despite the fact that the percentage of Open Resolvers class stayed steady with 9%, they became the second biggest resolvers of .nl ccTLD NSes because of the drop in cloud resolvers. This decrease is mostly because nowadays, nearly all the ISPs are also trying to provide cloud solutions for enterprises and even for individuals, which decreases the number of cloud users and increases ISP users.

On the other hand, hosting originated IP addresses also experienced a slight increase from 4% to 5% and likewise from IT Enterprises, Telecommunication and Research-based resolvers each one increased their shares in .nl ccTLD NSes more or less 1%.

The confidence interval analysis which run for March 20, 2019 is also run for May 22, 2019 data set. The results are shared in the Table 18 below.

| | *Number of Instances* | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **Confidence Interval Percentages** | *Cloud* | *Hosting* | *ISP* | *IT Enterprise* | *Research* | *Telecommunication* | *Open Resolvers* | *Total* |
| **[90-100]** | 0 | 3 | 53194 | 4142 | 1 | 0 | 11012 | 68352 |
| **[80,90)** | 2104 | 603 | 129947 | 3728 | 9 | 21 | 13704 | 150116 |
| **[70,80)** | 15973 | 2157 | 102878 | 5329 | 298 | 436 | 14470 | 141541 |
| **[60,70)** | 26604 | 8364 | 126492 | 24653 | 2284 | 9187 | 34907 | 232491 |
| **[0,60)** | 52494 | 45891 | 484040 | 56796 | 38019 | 47560 | 49334 | 774134 |
| *Total:* | 97175 | 57018 | 896551 | 94648 | 40611 | 57204 | 123427 | 1366634 |

Table 18: Total number of IP addresses for each confidence interval on May 22, 2019

Confidence on May dataset is considerably decreased for all the classes in comparison to March dataset. Despite the fact that open resolvers still have the highest percentage among other classes in terms of confidence interval greater than 90%, their share is decreased from 11.5% in March to 9% in May. Other classes which had acceptable percentages in high confidence intervals also decreased their shares in high confidence intervals to low confidence intervals. According to the table, ISPs are decreased from 6.5% to 6%, and IT Enterprises are decreased from 6.1% to 4.3% in terms of confidence interval greater than 90%.

On the other hand, classes who had more IPs in the lower confidence intervals, which are Cloud, Research, Hosting and Telecommunication did not change much. This is because they have already had fewer IP addresses in the high percentages of the confidence interval. There are slight changes in the number of IP addresses classified in each class when data sets from March and May are used. However, these changes are becoming less visible when interpreting the data with their inner percentages.

### 4.2.4 Monitoring Operational Changes on Day May 22, 2019

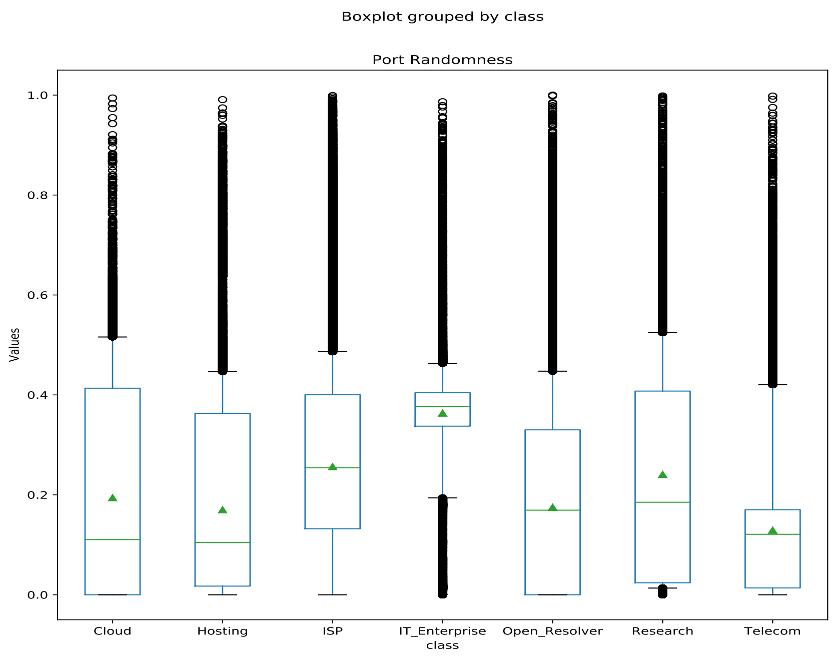


Figure 30: Box plot of port randomness feature amongst predicted classes on May 22, 2019

From March to May, there is a slight increase in ranges and means of port randomness for each class type. This increase can be seen in Figure 30. This is also supporting the claim that newer software generally meaning higher standard deviations, which was discussed in Section 3.4.9. The most notable change on port randomness is in ISP class. In the past two months, resolvers originating from ISPs are using a wider range of port numbers when contacting .nl NSes

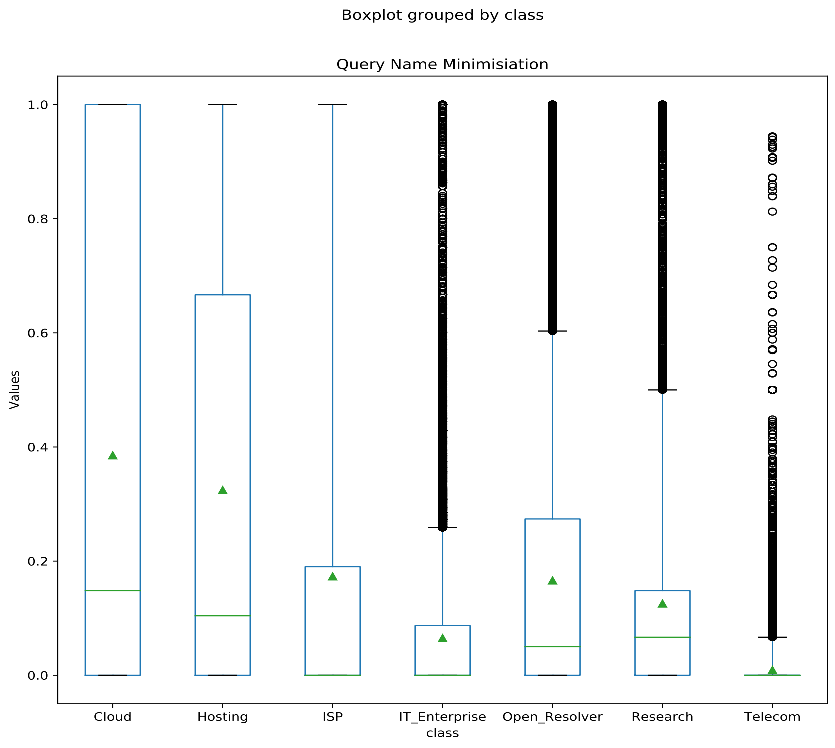


Figure 31: Box plot of qname minimisation feature amongst predicted classes on May 22, 2019

Slight improvements are also visible from Figure 31 on query name minimisation as well. Especially, resolvers from hosting and cloud origins have improved their qname minimisation compliant queries considerably. In telecommunication originated resolvers there is no observable change and IT Enterprise originating resolvers had a drop in the qname minimisation compliant queries.

The last analysis in the results section is “www.” usage in the queries on May 22, 2019. Overall in all class types, usage of “www.” in the queries are dropped slightly. This drop can be seen in Figure 32. This is an expected result to see a drop because the shares of qname minimisation are increased, which indirectly also affected “www.” usage. However, there is a slight increase in IT Enterprise related resolvers in “www.” usage. This is not an odd result because there was a slight decrease also in IT Enterprise related resolvers on qname minimisation and increase in “www.” usage is not surprising or unexpected when this is analysed.

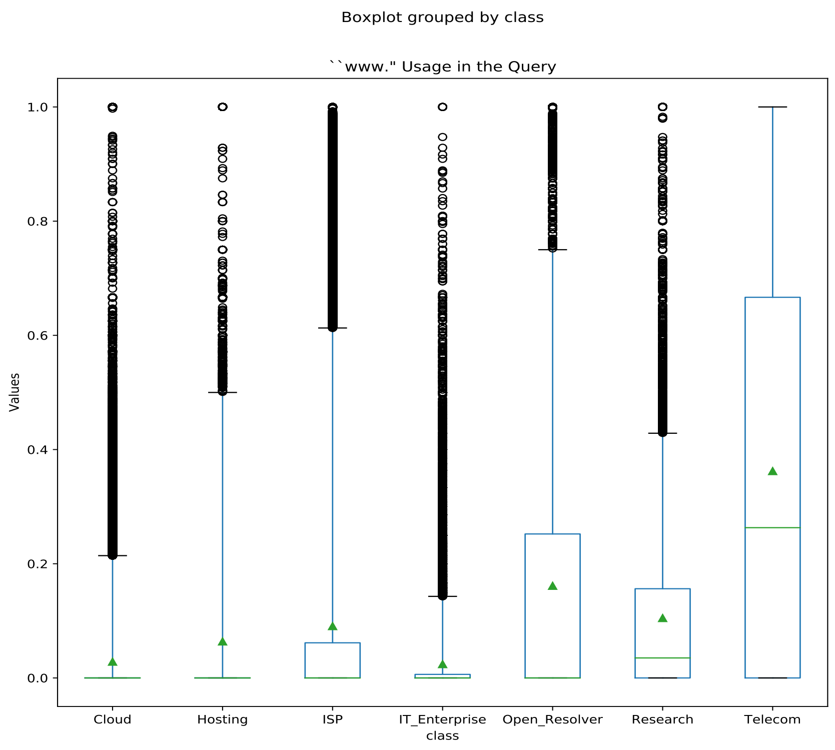


Figure 32: Box plot of “www.” usage feature amongst predicted classes on May 22, 2019

From the results on unlabelled data, change in query habits over time on different class types can be observed easily. This proves that this method can also be used for monitoring the operational changes over time on sector types. The key to do it successfully is to have a reliable ground truth and features which can represent the data on an angle that is useful for the point of the purpose.

As a conclusion, prediction statistics on March and May are following more or less the similar paths. This can be interpreted as the methodology used in this experiment is valid. Nonetheless, as DNS is a dynamic environment which has continuous operational changes and adaptations in even a month time, re-training the classifier with the new data can be useful for more precise results. Unfortunately, there is no study as similar to this study to be able to compare the results of these experiments.

# 5 Discussion & Conclusion

In this research, the main goal was to dig into the behaviours of recursive resolvers to profile them and then in the light of the appearing profile, to classify them according to their origins.

There were some steps that are followed to reach that goal. First of all, standard behaviours of the recursive resolvers and their behaviours in the wild explored. On top of it, necessary features that might be useful for the classification case are researched. These features should have been the features which could separate the recursive resolvers from one to other. Deciding the features were just the beginning of another research: ground truth data formation.

Different strategies are tried while forming the ground truth, but only the final method used for carrying out the rest of the research is explained in the paper, in Section 3.3. This was the most effective approach for ground truth formation out of all the other tested strategies. All the IP addresses are aggregated from their ASNs with a logical elimination approach which was discussed in Section 3.3.1, which their origins are labelled manually which can be seen in Appendix C.

Furthermore, after completing ground truth formation, a feature elimination process was applied to be able to reduce the dimensional of the data set to produce more efficiently classifiable data set without using too much computational resource.

As the final step, different machine learning approaches are followed to be able to pick the best one for classifying unlabelled data set. Random forest classifier was the best among all the other algorithms. Even though the accuracy was 91% for the random forest, the mistakes were, unfortunately, gathered on class types where ground truth data set was insufficient.

## 5.1 Limitations & Future Work

There were some constraints while conducting this research, which can be seen as follows:

- After spending six months on this research and looking into the related papers, there is still no specific scientific research on the subject of this thesis to be able to compare the results of the approach I have followed here.
- There is a lack of ground truth data in the field to be used for further improving the classifier's accuracy. As discussed in ground truth formation section, creating an accurate ground truth data is time-consuming, and even though a lot of time is spent on creating the most precise data for ground truth, it was not enough for all the target class types.
- Difference in accuracies of the ground truth data is leading to less efficient classification. To illustrate, open resolver dataset consists of nearly 100% accurate IP addresses, but this is not the case for research resolver IP addresses data set because the labelling of the sector is made on originating ASN numbers of the IP addresses.

- More collaboration is needed to be able to gather more accurate data. Local ISPs can be contacted for egress IP addresses of their resolvers, or known local hosting firms can also be contacted for the same reasoning.

In the end, I am not 100% sure of one of my motivations mentioned in the introduction: “the administrators of NSes would be able to understand which resolvers should be prioritised in case they are under a DDoS attack and have only limited resources to answer queries.” As can be seen from the results section, there still is a fuzziness in the classification, and even though 91% seems like a sufficient accuracy for prioritisation, in fact, it can affect hundreds of thousands of people in the case of millions of IP address classification.

As future work, the most important step is to find sufficient IP addresses from each class to represent classes better for the classifier. Next, as the values in the feature set define which class an IP address belongs to, new features can be tried to improve the accuracy of the classifier. The last point that can be improved is the type of classes provided in this research. The classes can be merged or extended to provide a better classification.

## 5.2 Conclusion

According to the research questions, following, it is explained how the answers are given.

- Research Question 1: “What is the expected behaviour of recursive resolvers?” This question is answered in the Literature Review in Section 2 in depth. The main reason this question is so crucial for the research was to be able to understand the behaviours of the recursive resolvers. Answers obtained from this question are further used while creating the features logically, which can represent the patterns of different recursive resolvers.

- Research Question 2: “How to classify the recursive resolvers at an authoritative name server?” To be able to answer this question, all the necessary steps for the machine learning case and different machine learning algorithms to classify recursive resolvers are covered in Sections 2.4 and 3.

- Research Question 3: “What are the main recursive resolvers of .nl NSes?” To be able to answer this, all the explained steps are applied to real DNS data, and the results which are answering this question is shared in Section 4. According to the results, the main recursive resolvers of .nl NSes are ISPs, Open Resolvers and Cloud resolvers.

As a conclusion, the research reached its goal; however, there are still points which can be improved starting from ground truth formation to re-evaluating all the feature set. This research is evaluated in the hope to open a door in other researchers’ mind to collect some attention on the subject and hopefully improve the DNS services served online.

## References

[1] How DNS Works. https://www.cloudflare.com/learning/dns/what-is-dns/. Accessed: August 16, 2019.

[2] Moritz Müller, Giovane Moura, Ricardo de O Schmidt, and John Heidemann. Recursives in the Wild: Engineering Authoritative DNS Servers. In *Proceedings of the 2017 Internet Measurement Conference*, pages 489–495. ACM, 2017.

[3] DNSSEC SIDN explanation. https://www.sidn.nl/faq/dnssec?language_id=2. Accessed: August 16, 2019.

[4] ENTRADA. http://entrada.sidnlabs.nl. Accessed: August 16, 2019.

[5] Algorithm Selection Cheat Sheet Scikit-Learn. https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html. Accessed: August 16, 2019.

[6] Box Plot Explanation. http://www2.ecology.su.se/dbhfj/2011gif/explanation.htm. Accessed: August 16, 2019.

[7] Wouter B. de Vriesa, Quirin Scheitle, Moritz Müller, Willem Toorop, Ralph Dolmans, and Roland van Rijswijk-Deij. A First Look at QNAME Minimization in the Domain Name System. In *Will be published*, pages –, 2018.

[8] Paul Mockapetris. RFC 1034: Domain names: Concepts and Facilities. *Status: Standard*, 6, 1987.

[9] Paul Mockapetris. RFC 1035: Domain Names: Implementation and Specification. *Status: Standart*, 1987.

[10] .NZ Registry Blog - Clustering . https://blog.nzrs.net.nz/source-address-clustering-feature-engineering/. Accessed: August 16, 2019.

[11] KSK Rollover Plan. https://www.iana.org/reports/2016/root-ksk-rollover-design-20160307.pdf. Accessed: August 16, 2019.

[12] IANA Root Hints. https://www.iana.org/domains/root/files. Accessed: August 16, 2019.

[13] How DNSSEC Works. https://www.internetsociety.org/resources/deploy360/2014/the-two-sides-of-dnssec-signing-and-validation/. Accessed: August 16, 2019.

[14] D Eastlake. Domain Name System Security Extensions. *RFC2535*, 1999.

[15] B Wellington. Domain name system security (DNSSEC) signing authority. Technical report, IETF, 2000.

[16] Brian Wellington. RFC 3007: Secure Domain Name System (DNS) Dynamic Update. Internet Engineering Task Force, November 2000.

[17] Marc Kührer, Thomas Hupperich, Jonas Bushart, Christian Rossow, and Thorsten Holz. Going wild: Large-scale Classification of Open DNS Resolvers.

In *Proceedings of the 2015 Internet Measurement Conference*, pages 355–368. ACM, 2015.

[18] David Dagon, Chris Lee, Wenke Lee, and Niels Provos. Corrupted DNS Resolution Paths: The Rise of a Malicious Resolution Authority. *Proc. 15th Network and Distributed System Security Symposium (NDSS), Internet Society, San Diego, CA (2008)*, 2008.

[19] Rami Al-Dalky and Kyle Schomp. Characterization of Collaborative Resolution in Recursive DNS Resolvers. In *International Conference on Passive and Active Network Measurement*, pages 146–157. Springer, 2018.

[20] Stephane Bortzmeyer. DNS Query Name Minimisation to Improve Privacy. Technical report, IETF, 2016.

[21] John R Koza, Forrest H Bennett, David Andre, and Martin A Keane. Automated design of both the topology and sizing of analog electrical circuits using genetic programming. In *Artificial Intelligence in Design'96*, pages 151–170. Springer, 1996.

[22] Andrew Ng. Machine learning and ai via brain simulations. *Accessed: May*, 3:2018, 2013.

[23] Thuy TT Nguyen and Grenville J Armitage. A survey of techniques for internet traffic classification using machine learning. *IEEE Communications Surveys and Tutorials*, 10(1-4):56–76, 2008.

[24] Matthew Roughan, Subhabrata Sen, Oliver Spatscheck, and Nick Duffield. Class-of-service mapping for qos: a statistical signature-based approach to ip traffic classification. In *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pages 135–148. ACM, 2004.

[25] Random Forrest Classifier in Scikit-Learn. `https://scikit-learn.org/stable/modules/ensemble.html#forests-of-randomized-trees`. Accessed: August 16, 2019.

[26] Leo Breiman. Random Forests. *Machine Learning*, 45(1):5–32, Oct 2001.

[27] Leo Breiman. Bagging Predictors. *Machine learning*, 24(2):123–140, 1996.

[28] Leo Breiman. *Classification and Regression Trees*. Routledge, 2017.

[29] Jaime Lynn Speiser, Michael E Miller, Janet Tooze, and Edward Ip. A Comparison of Random Forest Variable Selection Methods for Classification Prediction Modeling. *Expert Systems with Applications*, 2019.

[30] Thomas G Dietterich. Ensemble Methods in Machine Learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

[31] Karl Pearson F.R.S. X. on the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 50(302):157–175, 1900.

[32] C. R. Rao. *Karl Pearson Chi-Square Test The Dawn of Statistical Inference*, pages 9–24. Birkhäuser Boston, Boston, MA, 2002.

[33] Mary L McHugh. The Chi-Square Test of Independence. *Biochemia medica: Biochemia medica*, 23(2):143–149, 2013.

[34] Alper Kursat Uysal and Serkan Gunal. A Novel Probabilistic Feature Selection Method for Text Classification. *Knowledge-Based Systems*, 36:226–235, 2012.

[35] .NZ Registry Blog - Clustering . https://blog.nzrs.net.nz/detecting-resolvers-at-nz/. Accessed: August 16, 2019.

[36] Cristian Hesselman, Jelte Jansen, Maarten Wullink, Karin Vink, and Maarten Simon. A privacy framework for dns big data applications. *tech. rep.*, 2014.

[37] Stephane Bortzmeyer. Dns privacy considerations. *Status: Standard*, 2015.

[38] Difference Between Luminati Proxy Service and VPN. https://luminati.io/blog/difference-vpn-luminati. Accessed: August 16, 2019.

[39] Taejoong Chung, Roland van Rijswijk-Deij, Balakrishnan Chandrasekaran, David Choffnes, Dave Levin, Bruce M. Maggs, Alan Mislove, and Christo Wilson. A Longitudinal, End-to-End View of the DNSSEC Ecosystem. In *Proceedings of the USENIX Security Symposium (Security'17)*, Vancouver, BC, Canada, August 2017.

[40] About RIPE Atlas. https://atlas.ripe.net/about/. Accessed: August 16, 2019.

[41] Open Resolver Information. http://dns.measurement-factory.com/surveys/openresolvers.html. Accessed: August 16, 2019.

[42] OpenDNS Information. https://www.opendns.com/about/. Accessed: August 16, 2019.

[43] OpenDNS Resolver IP Information. https://www.opendns.com/network-map-data/. Accessed: August 16, 2019.

[44] Google Public DNS Information. lhttps://developers.google.com/speed/public-dns/docs/intro. Accessed: August 16, 2019.

[45] Quad9 Information. https://www.quad9.net/about/. Accessed: August 16, 2019.

[46] Who's Knocking? Profiling Recursive Resolvers at Authoritative Name Servers. https://blog.apnic.net/2019/06/10/whos-knocking-profiling-recursive-resolvers-at-authoritative-name-servers/. Accessed: August 16, 2019.

[47] Information on scikit-learn Library. https://scikit-learn.org/stable/supervised_learning.html#supervised-learning. Accessed: August 16, 2019.

[48] Resource Record Types. https://www.cisco.com/c/en/us/support/docs/ip/domain-name-system-dns/12684-dns-resource.html. Accessed: August 16, 2019.

[49] State of IPv6 Deployment 2018. https://www.internetsociety.org/resources/2018/state-of-ipv6-deployment-2018/. Accessed: August 16, 2019.

[50] Susan Thomson, Christian Huitema, Vladimir Ksinant, and Mohsen Souissi. DNS Extensions to Support IP Version 6. Technical report, RFC 1886, December, 1995.

[51] Rich Rosenbaum. Using the Domain Name System to Store Arbitrary String Attributes. *Status: Standard*, 1993.

[52] Resource Record Explanations. https://en.wikipedia.org/wiki/List_of_DNS_record_types. Accessed: August 16, 2019.

[53] Arnt Gulbrandsen and Levon Esibov. A DNS RR for Specifying the Location of Services (DNS SRV). *Status: Standard*, 2000.

[54] Roy Arends, Rob Austein, Matt Larson, Dan Massey, and Scott Rose. Resource Records for the DNS Security Extensions. Technical report, RFC 4034 (Proposed Standard), 2005.

[55] David Conrad. Indicating Resolver Support of DNSSEC, RFC3225. *Status: Standard*, 2001.

[56] J Martin Bland and Douglas G Altman. Statistics notes: Measurement error. *BMJ*, 312(7047):1654, 1996.

[57] Richard Lippmann, David Fried, Keith Piwowarski, and William Streilein. Passive Operating System Identification From TCP/IP Packet Headers. In *Workshop on Data Mining for Computer Security*, volume 40. Citeseer, 2003.

[58] Each Operating System's Standard TTL Values. https://subinsb.com/default-device-ttl-values/. Accessed: August 16, 2019.

[59] James Paugh, Paul O'Leary, Robert S Wilbourn, Thanh Nguyen, Iurii Iuzifovich, and Erik D Fears. Distinguishing Human-Driven DNS Queries From Machine-to-Machine DNS Queries, December 25 2018. US Patent App. 10/164,989.

[60] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[61] Information on scikit-learn's Univariate Feature Selection Methods. https://scikit-learn.org/stable/modules/feature_selection.html#univariate-feature-selection. Accessed: August 16, 2019.

[62] Information on scikit-learn's Tree Based Feature Selection Method. https://scikit-learn.org/stable/modules/feature_selection.html#tree-based-feature-selection. Accessed: August 16, 2019.

[63] J. A. Benediktsson, P. H. Swain, and O. K. Ersoy. Neural network approaches versus statistical methods in classification of multisource remote sensing data. *IEEE Transactions on Geoscience and Remote Sensing*, 28(4):540–552, July 1990.

[64] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

# Appendices

## A Columns of Used Database

Table 19: The fields of used database [4]

| column | protocol | type | description |
|---|---|---|---|
| id | DNS | query | message id |
| rcode | DNS | response | rcode (-1 is no matching server response is found) |
| opcode | DNS | query | opcode |
| query_ts | - | META | packet timestamp in UTC, uses TIMESTAMP datatype |
| unixtime | - | META | packet timestamp, seconds since January 1, 1970, UTC, BIGINT datatype |
| time | - | META | milliseconds since January 1, 1970, 00:00:00 UTC |
| qname | DNS | request | qname from request |
| qtype | DNS | request | qtype from request |
| domain name | DNS | META | secondlevel domainname (extracted from qname) |
| labels | DNS | META | count of the number of qname labels |
| src | IP | request | source IP address |
| dst | IP | request | destination IP address |
| ttl | IP | request | TTL |
| frag | IP | request | fragment count |
| ipv | IP | request | IP version, 4 or 6 |
| prot | IP | request | protocol, 6(TCP) or 17(UDP) |
| srcp | UDP/TCP | request | source port |
| dstp | UDP/TCP | request | destination port |
| udp_sum | UDP | request | checksum for the UDP request |
| dns_len | DNS | request | length dns request excluding ip/tcp/udp headers |
| dns_res_len | DNS | response | length dns response excluding ip/tcp/udp headers |
| len | DNS/UDP/TCP | request | length of the request packet including all headers |
| res_len | DNS/UDP/TCP | response | length of the response packet including all headers |
| aa | DNS | response header | Authoritative Answer |
| tc | DNS | response header | Truncation |
| rd | DNS | request header | Recursion Desired |
| ra | DNS | response header | Recursion Available |

Table 19: The fields of used database [4]

| column | protocol | type | description |
|---|---|---|---|
| z | DNS | request header | Zero |
| ad | DNS | response header | Authenticated data (DNSSEC) |
| cd | DNS | request header | Checking Disabled (DNSSEC) |
| ancount | DNS | response header | Answer Record Count |
| arcount | DNS | response header | Additional Record Count |
| nscount | DNS | response header | Authority Record Count |
| qdcount | DNS | request header | Question Count |
| country | IP | META | country location of the source IP address |
| asn | IP | META | autonomous system number of the source IP address |
| edns_udp | DNS | request | max UDP packet length supported by client |
| edns_version | DNS | request | EDNS0 version |
| edns_do | DNS | request | DNSSEC do-bit |
| edns_ping | DNS | request | EDNS0 ping option of powerdns |
| edns_nsid | DNS | request | name server identifier (rfc5001) |
| edns_dnssec_dau | DNS | request | DNSSEC Algorithm signalling, DNSSEC Algorithm Understood, (rfc6975) |
| edns_dnssec_dhu | DNS | request | DNSSEC Algorithm signalling, DS Hash Understoodd, (rfc6975) |
| edns_dnssec_n3u | DNS | request | DNSSEC Algorithm signalling, NSEC3 Hash Understood, (rfc6975) |
| edns_client_subnet | DNS | request | Client subnet option (draft-ietf-dnsop-edns-client-subnet-00) |
| edns_client_subnet_asn | - | META | asn of the client subnet |
| edns_client_subnet_country | - | META | country location of the client subnet IP address |
| edns_other | DNS | request | All other used EDNS0 options (concatenated as string) |

Table 19: The fields of used database [4]

| column | protocol | type | description |
|---|---|---|---|
| time_micro | - | META | the microseconds of the request timestamp (unixtime is rounded to seconds) |
| resp_frag | IP | request | the number of IP packet fragments required for the DNS response |
| proc_time | - | META | the number microseconds between the request and the response |
| is_google | - | META | true is the IP address matches one of the know Google resolver IP addresses |
| is_opendns | - | META | true is the IP address matches one of the know OpenDNS resolver IP addresses |
| server_location | META | request | location of the anycast node, only if anycast encoding is used for the file input directory |
| query_ts | DNS | request timestamp | request timestamp |
| edns_padding | DNS | request | Is EDNS0 Padding used |
| pcap_file | DNS | request | Name of the input pcap file |
| edns_keytag_count | DNS | request | number of EDNS0 keytags found |
| edns_keytag_list | DNS | request | EDNS0 keytags as comma separated list |
| q_tc | DNS | request | TC flag from request header |
| q_ra | DNS | request | RA flag from request header |
| q_ad | DNS | request | AD flag from request header |
| q_rcode | DNS | request | RCODE flag from request header |
| year | META | request | year part of timestamp |
| month | META | query | month part of timestamp |
| day | META | request | day part of timestamp |
| server | DNS | request | The name server the DNS request was sent to |

## B Columns of Ground Truth Forming Database

Table 20: Fields of ground truth forming database

| Column | Type | Explanation |
|---|---|---|
| unixtime | float | Unix Time Stamp |
| time | timestamp | Time Stamp |
| src | string | Source IP Addres |
| vp | string | Vantage Point |
| dst | string | Destination IP address |
| srv | string | Destination Server |
| query_id | string | Query's ID |
| qname | string | Query Name |
| len | int | Length of the packet |
| qtype | float | RR Type |
| ednsv | float | EDNS version |
| msm | string | Measurement Type |
| state | string | Measuremnt which is not relevant for this study |
| asn | float | ASN of Source IP |
| country | string | IP Address' Origin Country |
| year | string | Year |
| month | string | Month |
| day | string | Day |

## C ASNs of Ground Truth

| ASN | Class | Company |
|---|---|---|
| 51057 | Cloud | 42com |
| 16509 | Cloud | Amazon |
| 33873 | Cloud | arvato Systems perdata |
| 51401 | Cloud | arvato Systems perdata |
| 9166 | Cloud | Cegeka |
| 14061 | Cloud | DigitalOcean |
| 61098 | Cloud | EXOSCALE |
| 44066 | Cloud | first-colo.net |
| 34549 | Cloud | Germany (DE) Flag |
| 15725 | Cloud | https://www.iks-service.de |
| 13246 | Cloud | INETWIRE-AS |
| 39257 | Cloud | Interactive Network Communications |
| 24586 | Cloud | Intermax Cloudsourcing B.V. |
| 29791 | Cloud | Internap Corporation |
| 21100 | Cloud | ITL LLC |
| 58138 | Cloud | Korton Internet B.V. |
| 63949 | Cloud | linode |
| 34612 | Cloud | Matrix PC b.v. |
| 8075 | Cloud | Microsoft |
| 12676 | Cloud | NCORE |
| 23393 | Cloud | NuCDN LLC |
| 16276 | Cloud | OVH |
| 5521 | Cloud | PlusServer |
| 61157 | Cloud | PlusServer GmbH |
| 51862 | Cloud | Profitbricks (1&1) |
| 35003 | Cloud | qsc |
| 41696 | Cloud | Ram Mobile Data (Netherlands) B.V. |
| 8741 | Cloud | Ratiokontakt |
| 8823 | Cloud | Rockenstein AG |
| 8315 | Cloud | Sentia Netherlands |
| 36351 | Cloud | SoftLayer Technologies |
| 8208 | Cloud | Teamware GmbH |
| 48173 | Cloud | Unbelievable Machine Company GmbH |
| 28788 | Cloud | unilogic |
| 31673 | Cloud | uniserver |
| 205390 | Cloud | walter cloud services GmbH |
| 60955 | Cloud | Wavecon GmbH |
| 199860 | Cloud | xelent |
| 60144 | Hosting | 3winfra |
| 48344 | Hosting | 68a |
| 21476 | Hosting | all-connect Data Communications GmbH |
| 34788 | Hosting | all-inkl.com |
| 51430 | Hosting | AltusHost B.V. |

| ASN | Class | Company |
|---|---|---|
| 8893 | Hosting | ARTFILES |
| 48635 | Hosting | Astralus B.V. |
| 12657 | Hosting | BAYCIX |
| 20473 | Hosting | Choopa |
| 39704 | Hosting | CJ2 Hosting B.V |
| 58291 | Hosting | ColoCenter |
| 9182 | Hosting | Comspace |
| 51167 | Hosting | Contabo GmbH |
| 20849 | Hosting | CONTINUM A.G. |
| 44716 | Hosting | D-hosting die Rackspace & Connectivity GmbH |
| 44946 | Hosting | Dembach Goo Informatik GmbH & Co. KG |
| 8763 | Hosting | DENIC |
| 25542 | Hosting | DENIT |
| 203329 | Hosting | e-shelter services |
| 42730 | Hosting | evanzo |
| 49024 | Hosting | fhe3 |
| 60351 | Hosting | Geib-it |
| 12586 | Hosting | GHOSTnet |
| 48918 | Hosting | globalways |
| 20773 | Hosting | GoDaddy |
| 201709 | Hosting | Hanse Datacenter Services GmbH |
| 24940 | Hosting | Hetzner |
| 196922 | Hosting | Hofmeir Media GmbH |
| 8972 | Hosting | hosteurope |
| 59795 | Hosting | hosting4real |
| 29140 | Hosting | Hostserver |
| 24679 | Hosting | Hostway Deutschland GmbH |
| 15368 | Hosting | Intares GmbH |
| 20507 | Hosting | InterNLnet |
| 9135 | Hosting | ITEMAX |
| 196763 | Hosting | Key-systems |
| 31103 | Hosting | Keyweb |
| 8391 | Hosting | Knipp |
| 28753 | Hosting | LeaseWeb |
| 60781 | Hosting | LeaseWeb |
| 42699 | Hosting | managedhosting.de GmbH |
| 62310 | Hosting | managedhosting.de GmbH |
| 34240 | Hosting | manitu GmbH |
| 43341 | Hosting | MDLink |
| 35833 | Hosting | mpex GmbH |
| 24961 | Hosting | myLoc managed IT AG |
| 43847 | Hosting | nbiserv |
| 25459 | Hosting | NedZone Internet BV |
| 15743 | Hosting | net.de AG |
| 197540 | Hosting | netcup |
| 57342 | Hosting | netznutz.net |

| ASN | Class | Company |
|---|---|---|
| 199938 | Hosting | netzwerge |
| 43350 | Hosting | nforce NL |
| 9067 | Hosting | one4vision GmbH |
| 48200 | Hosting | opteamax |
| 43366 | Hosting | osso nL |
| 12306 | Hosting | PLUSLINE |
| 49855 | Hosting | plutex |
| 29018 | Hosting | smartTERRA GmbH |
| 6724 | Hosting | STRATO AG |
| 33984 | Hosting | surfplanet |
| 25291 | Hosting | SysEleven GmbH |
| 20857 | Hosting | TRANSIP |
| 29066 | Hosting | Velia.net Internetdienste GmbH |
| 15535 | Hosting | Virtual Access Internet BV |
| 21413 | ISP | ENVIA-TEL |
| 21221 | ISP | INFOPACT |
| 24603 | ISP | XOO |
| 3265 | ISP | XS4ALL |
| 30766 | ISP | http://www.ggew-net.de |
| 15435 | ISP | caiway.nl |
| 198570 | ISP | .stadtnetz-bamberg |
| 8881 | ISP | 1&1 Versatel DE |
| 25038 | ISP | Alfred Schruff trading as AIXTRANET |
| 1200 | ISP | Amsterdam Internet Exchange |
| 12392 | ISP | Asbrutele Voo |
| 2611 | ISP | Belnet |
| 20686 | ISP | BISPING |
| 12859 | ISP | BIT BV |
| 198967 | ISP | Bitel DE |
| 20886 | ISP | bn-online.net |
| 2110 | ISP | BT Ireland |
| 25596 | ISP | Cambrium IT Services B.V. |
| 12732 | ISP | Carrier51 GmbH GutCon GmbH |
| 25058 | ISP | CMO Internet Dienstleistungen GmbH |
| 174 | ISP | cogentco |
| 8220 | ISP | COLT Technology Services Group Limited |
| 30962 | ISP | comtrance |
| 34154 | ISP | configo |
| 5419 | ISP | Cubic Circle |
| 3320 | ISP | DE Telecom |
| 15542 | ISP | Delta NL |
| 60294 | ISP | Deutsche Glasfaser Wholesale |
| 15763 | ISP | DOKOM Gesellschaft fuer Telekommunikation mbH |
| 12312 | ISP | ecotel |
| 9031 | ISP | EDPNET |
| 41585 | ISP | ELEMENTMEDIA GmbH |

| ASN | Class | Company |
|---|---|---|
| 5390 | ISP | EuroNET |
| 13237 | ISP | European Backbone of AS13237 |
| 9145 | ISP | ewetel.de |
| 47215 | ISP | filoo |
| 44194 | ISP | Foerderverein Freie Netzwerke |
| 5430 | ISP | freenet Datenkommunikations |
| 64475 | ISP | Freifunk Frankfurt am Main e.V. |
| 49009 | ISP | Freifunk Rheinland e.V. |
| 201701 | ISP | freifunk-rheinland |
| 24956 | ISP | Gaertner Datensysteme GmbH & Co. KG |
| 9063 | ISP | VSE NET |
| 12355 | ISP | Helinet |
| 50469 | ISP | hessenkom |
| 5580 | ISP | Hibernia Networks |
| 50324 | ISP | hofnetz |
| 13045 | ISP | HTP |
| 31400 | ISP | http://www.accelerated.de |
| 6939 | ISP | Hurricane Electric |
| 16218 | ISP | IACD Autonomous System |
| 29670 | ISP | Individual Network Berlin e.V. |
| 42652 | ISP | inexio Informationstechnologie und Telekommunikation |
| 13030 | ISP | Init7 (Switzerland) Ltd |
| 5669 | ISP | Interoute |
| 8928 | ISP | INTEROUTE |
| 12941 | ISP | intersaar |
| 20647 | ISP | IPB Internet Provider in Berlin GmbH |
| 12731 | ISP | IPHH Internet Port Hamburg GmbH |
| 198089 | ISP | ipvisie |
| 198818 | ISP | ispeg |
| 29624 | ISP | iwelt DE |
| 202329 | ISP | karlsruhe freifunk |
| 29413 | ISP | Komro |
| 286 | ISP | KPN |
| 1136 | ISP | KPN |
| 8737 | ISP | KPN |
| 12871 | ISP | KPN B.V. |
| 47377 | ISP | KPN BE |
| 15879 | ISP | KPN Internedservices B.V. |
| 3356 | ISP | Level3 LLC US |
| 12374 | ISP | LF.net |
| 6830 | ISP | Liberty Global |
| 16298 | ISP | Lubbers Box Telematica BV |
| 9009 | ISP | M247 |
| 8365 | ISP | Man.da |
| 25394 | ISP | MK Netzdienste GmbH & Co. KG |
| 8767 | ISP | MNET |

| ASN | Class | Company |
|---|---|---|
| 31477 | ISP | MNT-DUOCAST |
| 12502 | ISP | NEPUSTILNET |
| 8422 | ISP | NetCologne |
| 20810 | ISP | Netcom Kassel |
| 8319 | ISP | NETHINKS |
| 49784 | ISP | Netvisit B.V. |
| 207176 | ISP | OpenFiber |
| 56912 | ISP | OR Network |
| 8859 | ISP | OSN |
| 54825 | ISP | Packet Host |
| 20676 | ISP | Plusnet |
| 20783 | ISP | pop-interactive GmbH |
| 15987 | ISP | PORTUNITY |
| 8687 | ISP | PPP |
| 16097 | ISP | Pyur.com |
| 203228 | ISP | RHOENNET |
| 28685 | ISP | Routit BV |
| 29014 | ISP | ScaleUp Technologies GmbH & Co. Kg |
| 6735 | ISP | sdt.net AG |
| 12414 | ISP | Solcon |
| 5539 | ISP | SpaceNET AG |
| 15657 | ISP | Speedbone Internet & Connectivity GmbH |
| 12431 | ISP | SYBCOM |
| 50266 | ISP | T-Mobile |
| 31615 | ISP | T-Mobile |
| 8820 | ISP | TAL DE |
| 58243 | ISP | Tele AG |
| 16202 | ISP | Tele Columbus |
| 20880 | ISP | Tele Columbus |
| 35244 | ISP | Tele Columbus AG |
| 13127 | ISP | Tele2 |
| 21263 | ISP | TeleData |
| 6805 | ISP | Telefonica Germany |
| 6848 | ISP | Telenet |
| 59507 | ISP | terralink |
| 199284 | ISP | Thueringer Netkom |
| 42184 | ISP | tkrz Stadtwerke GmbH |
| 16316 | ISP | TMT GmbH & Co. KG |
| 13101 | ISP | TNG Stadtnetz GmbH |
| 5409 | ISP | TopLink DE |
| 15844 | ISP | True global communications GmbH |
| 21385 | ISP | Trusted Network GmbH |
| 29396 | ISP | Unet Network, The Netherlands |
| 29562 | ISP | Unitymedia BW GmbH |
| 62336 | ISP | Upstream |
| 34372 | ISP | VegaSystems GmbH |

| ASN | Class | Company |
|---|---|---|
| 702 | ISP | Verizon |
| 6753 | ISP | Viprinet Europe GmbH |
| 1273 | ISP | Vodafone |
| 3209 | ISP | Vodafone DE |
| 31334 | ISP | Vodafone Kabel Deutschland GmbH |
| 33915 | ISP | Vodafone Libertel B.V |
| 15480 | ISP | Vodafone NL Autonomous System |
| 15943 | ISP | wilhem.Tel |
| 9136 | ISP | Wobcom |
| 42 | ISP | WoodyNet |
| 9211 | ISP | WORK DE |
| 15426 | ISP | XENOSITE |
| 12480 | ISP | ilk.net |
| 39637 | IT Firm | ADES BV |
| 35258 | IT Firm | akquinet outsourcing gGmbH |
| 31317 | IT Firm | AnschlussWerk GmbH |
| 48972 | IT Firm | betterbe |
| 43996 | IT Firm | Booking.com |
| 202196 | IT Firm | Booking.com |
| 57407 | IT Firm | CK Software |
| 30823 | IT Firm | combahton |
| 33824 | IT Firm | consol |
| 25504 | IT Firm | CRONON |
| 42707 | IT Firm | E-quest |
| 205597 | IT Firm | FROST |
| 202077 | IT Firm | geneon |
| 20677 | IT Firm | imos.net |
| 34171 | IT Firm | Inter.Net Germany |
| 13289 | IT Firm | iWelt AG |
| 197729 | IT Firm | keyidentity |
| 31259 | IT Firm | komsa |
| 203412 | IT Firm | KUES DATA GmbH |
| 15894 | IT Firm | LEITWERK |
| 39063 | IT Firm | leitwert |
| 24764 | IT Firm | m.a.x. Informationstechnologie AG |
| 48821 | IT Firm | mauve |
| 200567 | IT Firm | mdex |
| 205614 | IT Firm | Medialine EuroTrade AG |
| 20755 | IT Firm | NET-LAB |
| 36236 | IT Firm | netactuate |
| 200519 | IT Firm | nynex |
| 12348 | IT Firm | odn.de |
| 15945 | IT Firm | PFALZKOM-NET |
| 21473 | IT Firm | PFALZKOM-NET |
| 33988 | IT Firm | proact |
| 29686 | IT Firm | Probe Networks |

| ASN | Class | Company |
|---|---|---|
| 41887 | IT Firm | Prolocation |
| 16188 | IT Firm | punkt |
| 205617 | IT Firm | qsit |
| 34928 | IT Firm | regio iT aachen GmbH |
| 198203 | IT Firm | routelabel.net |
| 39702 | IT Firm | S-IT Informationstechnologie Betreiber GmbH & Co. KG |
| 202040 | IT Firm | SCT Schiele GmbH |
| 41955 | IT Firm | sernet |
| 50673 | IT Firm | Serverius Holding B.V. |
| 198726 | IT Firm | smartservice |
| 202625 | IT Firm | softtech |
| 47297 | IT Firm | Telekommunikation Lindau (B) GmbH |
| 29432 | IT Firm | TREX Regional Exchanges Oy |
| 43566 | IT Firm | tyntec |
| 7342 | IT Firm | Verisign |
| 396560 | IT Firm | Verisign |
| 396561 | IT Firm | Verisign |
| 396562 | IT Firm | Verisign |
| 397208 | IT Firm | Verisign |
| 48921 | IT Firm | Verotel International B.V. |
| 51978 | IT Firm | WEMACOM Telekommunikation |
| 48111 | IT Firm | westnetz |
| 44156 | IT Firm | winnen |
| 39647 | IT Firm | Within Reach Group B.V. |
| 58075 | IT Firm | X2com |
| 50343 | IT Firm | xing |
| 203969 | IT Firm | YMC AG |
| 43910 | IT Firm | intra2net |
| 12923 | IT Firm | Wizard Computersysteme GmbH |
| 47610 | Research | Aachen Uni |
| 2614 | Research | AARNIEC (Romania) |
| 250 | Research | AS250 Foundation |
| 553 | Research | BelWue-Koordination |
| 56357 | Research | Chair for Network Architectures and Services |
| 50472 | Research | Chaos Computer Club |
| 6766 | Research | Chaos Darmstadt (Hacking) |
| 8283 | Research | coloclue.net |
| 205046 | Research | FZI Forschungszentrum Informatik am Karlsruher Institut fuer Technologie |
| 50595 | Research | Hochschule RheinMain, University of Applied Sciences, Wiesbaden Ruesselsheim Geisenheim |
| 2857 | Research | Johannes Gutenberg-Universitaet |
| 34878 | Research | Karlsruhe Institute of Technology |
| 12816 | Research | Leibniz Rechenzentrum |
| 49697 | Research | netshelter |
| 3333 | Research | Ripe NCC |

| ASN | Class | Company |
|---|---|---|
| 29484 | Research | Ruhr-Universitaet Bochum |
| 1140 | Research | SIDN |
| 1101 | Research | Surfnet |
| 1102 | Research | Surfnet |
| 1103 | Research | Surfnet |
| 1133 | Research | Surfnet |
| 680 | Research | Verein zur Foerderung eines Deutschen Forschungsnetzes |
| 47447 | Telecominication | 23media GmbH |
| 25054 | Telecominication | ACO Computerservice GmbH |
| 30925 | Telecominication | CBizz B.V. |
| 39151 | Telecominication | Ce-tel |
| 28876 | Telecominication | dacor |
| 201213 | Telecominication | DARZ GmbH |
| 43437 | Telecominication | Digifoon Group BV |
| 29259 | Telecominication | IABG Teleport GmbH |
| 15372 | Telecominication | IBH IT-Service GmbH |
| 33808 | Telecominication | itenos |
| 29252 | Telecominication | NetCom BW GmbH |
| 41998 | Telecominication | NetCom BW GmbH |
| 15594 | Telecominication | NETZQUADRAT |
| 15415 | Telecominication | Oberberg-Online Informationssysteme GmbH |
| 12611 | Telecominication | R-KOM Regensburger Telekommunikations GmbH & Co. KG |
| 60316 | Telecominication | RS Gesellschaft fuer Informationstechnik mbH & Co.KG |
| 51417 | Telecominication | tbits |
| 12843 | Telecominication | telemaxx |
| | | |

Table 21: ASN to class mappings of Ripe and Luminati measurements