Network Traffic Analysis and Machine Learning

Outline

- Introduction
- Network Defence
- Machine learning

I.Introduction

- Network security is the broad practice of protecting computer networks and network-accessible endpoints from *malice, misuse,* and *denial*.
- Firewalls are perhaps the best-known network defense systems,
  - Enforcing access policies and filtering unauthorized traffic between artifacts in the network.
- Network defense is about more than just firewalls

The OSI Model

Throughout this chapter, we refer to different parts of a typical networking stack using the well-known Open Systems Interconnection (OSI) model. The OSI model contains seven layers (see also Figure 5-1):

Layer 1: physical layer
: Converts digital data to electrical or mechanical signals that can be transmitted over the network, and converts signals received over the network to digital data

Layer 2: data link layer
: Drives data transfer between adjacent nodes in a physical network

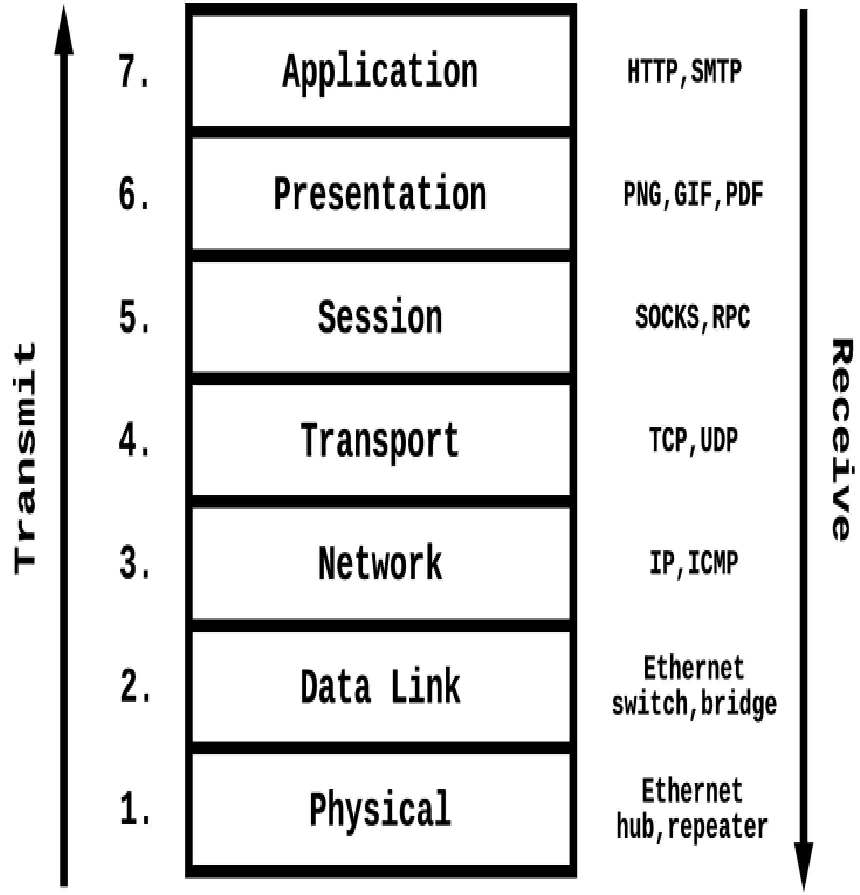
Layer 3: network layer
: Routes packets and performs flow control between two points on a network

Layer 4: transport layer
: Provides host-to-host communication, dictates the quality and reliability of data transmission

Layer 5: session layer
: Initiates, maintains, and closes sessions between application processes

Layer 6: presentation layer
: Translates binary data into formats understandable by applications

Layer 7: application layer
: Displays data received from the network to users

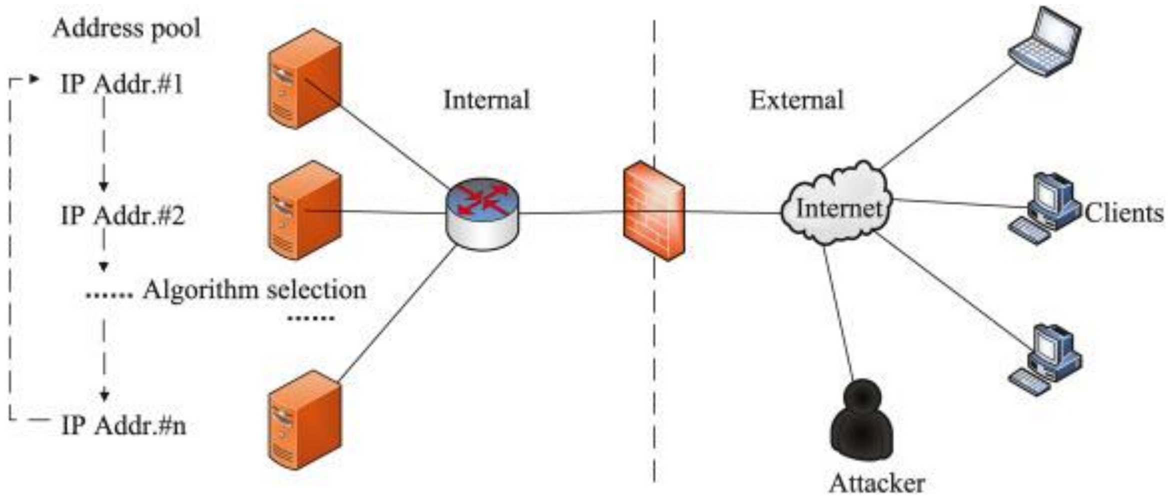


Introduction

- Context:
  - we look at techniques for classifying network traffic
- Our discussion of network security is limited to packet-based information transmission.
  - Each packet is transmitted over the network layer and formatted in an appropriate protocol by the transport layer, with the *reconstruction of the information stream* from individual packets occurring at the session layer or above.
- Focus: The security of the *network*, *transport*, and *session* layers (layers 3, 4, and 5 of the OSI model, respectively)

II.Theory of Network Defense

- Networks have a complicated defense model
- That is because of the broad range of attack surfaces and threat vectors



Access Control and Authentication

- **Access control:**
  - a form of authorization by which you can control which users, roles, or hosts in the organization can access each segment of the network.
- **Firewalls**:
  - a means of access control:
  - enforcing predefined policies for how hosts in the network are allowed to communicate with one another


```
# ACCEPT inbound TCP connections from 192.168.100.0/24 to port 22
iptables --append INPUT --protocol tcp --source 192.168.100.0/24
         --dport 22 --jump ACCEPT

# DROP all other inbound TCP connections to port 22
iptables --append INPUT --protocol tcp --dport 22 --jump DROP
```

Access Control and Authentication

- Active authentication methods gather more information about the connecting client,
  - often using *cryptographic methods, private knowledge,* or *distributed mechanisms* to achieve more reliable client identity attestation
- In some cases, *multifactor authentication* (MFA) can be a suitable method for raising the bar for attackers wanting to break in

Intrusion Detection

- Intrusion detection systems go beyond the initial access control barriers to detect attempted or successful breaches of a network by making passive observations.
- Intrusion prevention systems: intercept the direct line of communication between the source and destination and automatically act on detected anomalies.
- Real-time *packet sniffing* is a requirement for any intrusion detection or prevention system.

Detecting In-Network Attackers

- Assuming that attackers can get past access control mechanisms and avoid detection by intrusion detection systems,
  - they will have penetrated the network perimeter and be operating within the trusted bounds of your infrastructure.
- Well-designed systems should assume that this is always the case and work on detecting in-network attackers
- Proper segmentation of a network can help limit the damage caused by in-network attackers.

Data-Centric Security

- Data-centric security emphasizes the security of the data itself, meaning that even if a database is breached, the data might not be of much value to an attacker.
- Encrypting stored data is a way to achieve data-centric security,
  - It has been a standard practice to salt and hash stored passwords so that attackers cannot easily make use of stolen credentials to take over user accounts

Honeypots

- Honeypots are decoys intended for gathering information about attackers
- Honeypots present interfaces that mimic the real systems, and
  - can sometimes be very successful in tricking attackers into revealing characteristics of their attack that can help with offline data analysis or active countermeasures

III. Machine Learning and Network Security

- *Pattern mining* is one of the primary strengths of machine learning, and there are many inherent patterns to be discovered in network traffic data.
- At first glance, the data in network packet captures might seem sporadic and random, but most communication streams follow a strict network protocol
- We can also find malicious activity in networks by mining for patterns and drawing correlations in the data, especially for attacks that rely on volume and/or iteration such as network scanning and denial of service (DoS) attacks

No.	Time	Source	Protocol	Destination	Length	Info
1	0.000…	192.168.1.104	TCP	216.18.166.136	74	49859 → 80 [SYN] Seq=0 Win=8192 Len…
2	0.307…	216.18.166.136	TCP	192.168.1.104	74	80 → 49859 [SYN, ACK] Seq=0 Ack=1 W…
3	0.307…	192.168.1.104	TCP	216.18.166.136	66	49859 → 80 [ACK] Seq=1 Ack=1 Win=17…

Figure 5-2. TCP three-way handshake (source: Wireshark screen capture)

From Captures to Features

- Capturing live network data is the primary way of recording network activity for online or offline analysis.
- packet analyzers (also known as packet/network/protocol snifers) intercept and log traffic in the network
- With access to information-rich raw data, the next step will be to generate useful features for data analysis.

From Captures to Features

- *tcpdump* is a command-line packet analyzer that is ubiquitous in modern Unix-like operating systems.
- Captures are made in the libpcap file format (.pcap),
  - which is a fairly universal and portable format for the captured network data

1.tcpdump

- A powerful tool that allows you to capture, parse, filter, decrypt, and search through network packets
- Example:
- These three packets were sent between the home/private IP address 192.168.0.112 and a remote HTTP server at IP address 93.184.216.34

```
# tcpdump -i any -c 3 tcp

3 packets captured
3 packets received by filter
0 packets dropped by kernel

12:58:03.231757 IP (tos 0x0, ttl 64, id 49793, offset 0,
        flags [DF], proto TCP (6), length 436)
    192.168.0.112.60071 > 93.184.216.34.http: Flags [P.],
cksum 0x184a (correct), seq 1:385, ack 1, win 4117,
options [nop,nop,TS val 519806276 ecr 1306086754],
length 384: HTTP, length: 384
    GET / HTTP/1.1
    Host: www.example.com
    Connection: keep-alive
```

2.Wireshark

- Wireshark is a capable alternative that provides a graphical user interface and has some additional features.
- It supports the standard libpcap file format, but by default captures packets in the PCAP Next Generation (.pcapng) format.
- Wireshark also provides a very simple interface that automatically converts all encrypted packets when you provide the private key and encryption scheme

3. extracted features

- Session duration: 4.971 secs
- Total session packets: 10
- Protocol: TCP
- Total bytes from source to destination:

62 + 54 + 616 + 54 + 54 + 54 = 894 bytes

- Total bytes from destination to source:

62 + 887 + 60 + 60 = 1069 bytes

- Successful TCP handshake: true
- Network service on the destination: HTTP
- Number of ACK packets: 4

No.	Time	Source	Destinat	Protocol	Lengt	Info
1	0.000…	192…	76.7…	TCP	62	1315 → 80 [SYN] Seq=0 Win=32767 Len=0 MSS=1460 SACK_PERM=1
2	0.349…	76.…	192.…	TCP	62	80 → 1315 [SYN, ACK] Seq=0 Ack=1 Win=16384 Len=0 MSS=1460 SACK_PERM=1
3	0.349…	192…	76.7…	TCP	54	1315 → 80 [ACK] Seq=1 Ack=1 Win=32767 Len=0
4	0.353…	192…	76.7…	HTTP	616	GET /exploit.php?id=6216 HTTP/1.1
5	0.764…	76.…	192.…	HTTP	887	HTTP/1.1 200 OK (text/html)
6	0.898…	192…	76.7…	TCP	54	1315 → 80 [ACK] Seq=563 Ack=834 Win=31934 Len=0
7	4.675…	192…	76.7…	TCP	54	1315 → 80 [FIN, ACK] Seq=563 Ack=834 Win=31934 Len=0
8	4.970…	76.…	192.…	TCP	60	80 → 1315 [ACK] Seq=834 Ack=564 Win=17424 Len=0
9	4.971…	76.…	192.…	TCP	60	80 → 1315 [FIN, ACK] Seq=834 Ack=564 Win=17424 Len=0
…	4.971…	192…	76.7…	TCP	54	1315 → 80 [ACK] Seq=564 Ack=835 Win=31934 Len=0

Figure 5-3. Attacker's TCP session (source: Wireshark screen capture)

4. extracted features

- Aggregating patterns across large sequences of packets can allow you to generate more useful information from the data than analyzing single packets

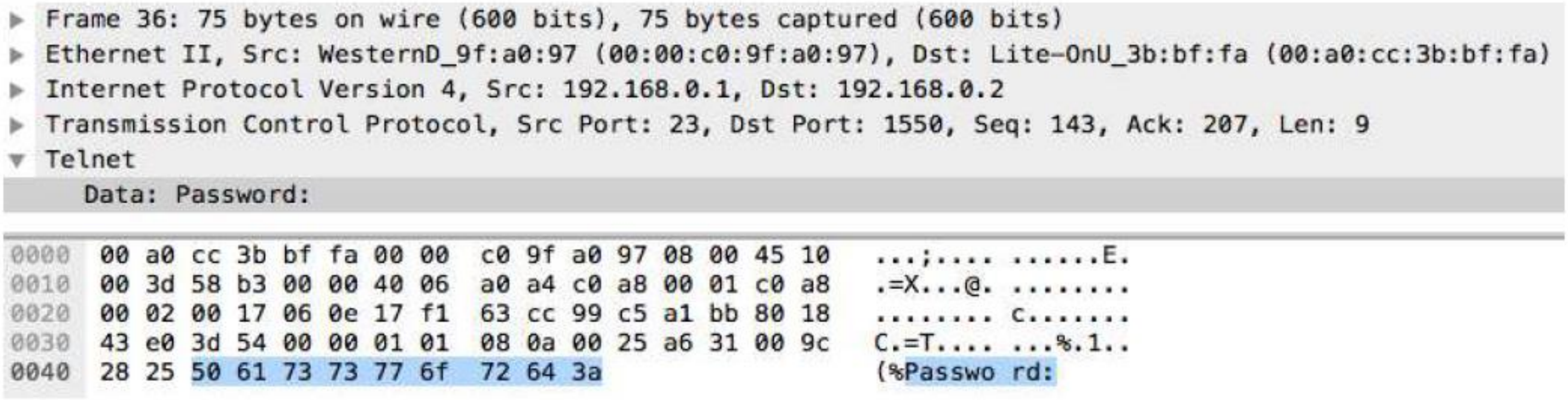


Figure 5-4. Data section of Telnet packet (source: Wireshark screen capture)

Some examples of features

- Application protocol (e.g., Telnet, HTTP, FTP, or SMTP)
- Encrypted
- Failed login attempt
- Successful login attempt
- Root access attempted (e.g., su root command issued)
- Root access granted
- Is guest login
- curl/wget command attempted
- File creation operation made

Threats in the Network

- We broadly categorize threats into passive and active attacks
- Further break down active attacks into four classes:
  - breaches,
  - spoofing,
  - pivoting (“lateral movement”), and
  - denial of service (DoS).

Passive attacks

- They do not
  - initiate communication with nodes in the network and
  - interact with or
  - modify network data.
- Attackers typically use passive techniques for information gathering and reconnaissance activity.
- Port scanning:
  - a passive network attack that bad actors use to probe for open ports to identify services running on servers

Active attacks: breaches

- Network breaches are perhaps the most prolific network attacks.
- The term “breach” can refer to either :
    - a hole in the internal network’s perimeter or
    - the act of an attacker exploiting such a hole

  to gain unauthorized access to private systems.
- Attackers can force their way into networks after performing passive information gathering and reconnaissance,
    - finding vulnerabilities in publicly accessible endpoints that allow them shell or root access to systems

Active attacks: Spoofing

- Attackers use spoofing (i.e., sending falsified data):
  - a mechanism for installing their presence in the middle of a trusted communications channel between two entities.
- DNS spoofing and ARP spoofing (aka *cache poisoning*) misuse network caching mechanisms to force the client to engage in communications with a spoofed entity instead of the intended entity

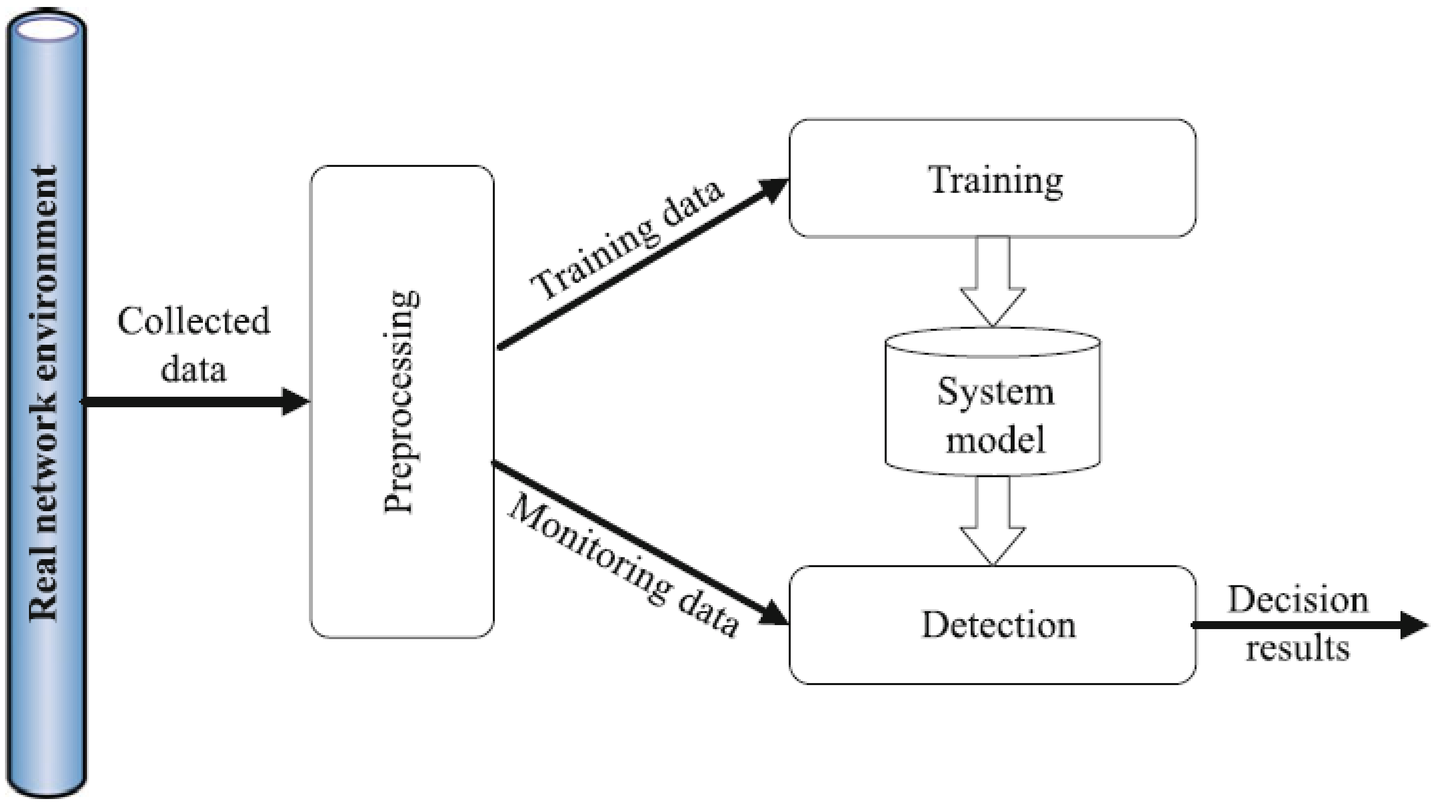
Active attacks: Pivoting

- Pivoting:
  - the technique of moving between servers in a network after an attacker has gained access to an entry point.

Active attacks: DoS

- Denial-of-service (DoS) attacks target the general availability of a system, disrupting access to it by intended users.
- There are many flavors of DoS attacks:
  - Including: the distributed DoS (DDoS) attack, which refers to the use of multiple IP addresses that might span a large range of geolocations to attack a service.

IV. Building a Predictive Model to Classify Network Attacks



Building a Predictive Model to Classify Network Attacks

- The dataset that we will use is the NSLKDD dataset,
  - which is an improvement to a classic network intrusion detection dataset used widely by security data science professionals.
- How:
  - The data was collected over nine weeks and consists of raw tcpdump traffic in a local area network (LAN) that simulates the environment of a typical United States Air Force LAN.
- Some network attacks were deliberately carried out during the recording period.

Building a Predictive Model to Classify Network Attacks

- Obtaining good training data is a perennial problem when using machine learning for security.
- Classifiers are only as good as the data used to train them,
  - Reliably labeled data is especially important in supervised machine learning

Building a Predictive Model to Classify Network Attacks

- Transfer:
  - With no good way to generate training data originating from the same source as the test data,
    - another alternative is to train the classifier on a comparable dataset, often obtained from another source or an academic study
  - Transfer learning, or inductive transfer:
    - the process of taking a model trained on one dataset and then customizing it for another related problem

Dataset: the NSLKDD dataset

- There were 38 different types of attacks, but only 24 are available in the training set.
- These attacks belong to four general categories:
  - Dos: Denial of service
  - R2l: Unauthorized accesses from remote servers
  - U2r: Privilege escalation attempts
  - Probe: Brute-force probing attacks

Exploring the Data: CSV file

- The last value in each CSV record is an artifact of the NSL-KDD improvement that we can ignore.
- The class label is the second-to-last value in each record, and the other 41 values correspond to these features

```
0,tcp,ftp_data,SF,491,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,
0.00,0.00,1.00,0.00,0.00,150,25,0.17,0.03,0.17,0.00,0.00,0.00,0.05,
0.00,normal,20

0,icmp,eco_i,SF,8,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,21,0.00,0.00,
0.00,0.00,1.00,0.00,1.00,2,60,1.00,0.00,1.00,0.50,0.00,0.00,0.00,
0.00,ipsweep,17
```

1	duration	9	urgent	17	num_file_creations	30	diff_srv_rate
2	protocol_type	10	hot	18	num_shells	31	srv_diff_host_rate
3	service	11	num_failed_logins	19	num_access_files	32	dst_host_count
4	flag	12	logged_in	20	num_outbound_cmds	33	dst_host_srv_count
5	src_bytes	13	num_compromised	21	is_host_login	34	dst_host_same_srv_rate
6	dst_bytes	14	root_shell	22	is_guest_login	35	dst_host_diff_srv_rate
7	land	15	su_attempted	23	count	36	dst_host_same_src_port_rate
8	wrong_fragment	16	num_root	24	srv_count	37	dst_host_srv_diff_host_rate
				25	serror_rate	38	dst_host_serror_rate
				26	srv_serror_rate	39	dst_host_srv_serror_rate
				27	rerror_rate	40	dst_host_rerror_rate
				28	srv_rerror_rate	41	dst_host_srv_rerror_rate
				29	same_srv_rate		

Here's what we find upon inspecting the contents of `category`:

```
{
    'benign': ['normal'],
    'probe':  ['nmap', 'ipsweep', 'portsweep', 'satan',
               'mscan', 'saint', 'worm'],
    'r2l':    ['ftp_write', 'guess_passwd', 'snmpguess',
               'imap', 'spy', 'warezclient', 'warezmaster',
               'multihop', 'phf', 'imap', 'named', 'sendmail',
               'xlock', 'xsnoop', 'worm'],
    'u2r':    ['ps', 'buffer_overflow', 'perl', 'rootkit',
               'loadmodule', 'xterm', 'sqlattack', 'httptunnel'],
    'dos':    ['apache2', 'back', 'mailbomb', 'processtable',
               'snmpgetattack', 'teardrop', 'smurf', 'land',
               'neptune', 'pod', 'udpstorm']
}
```

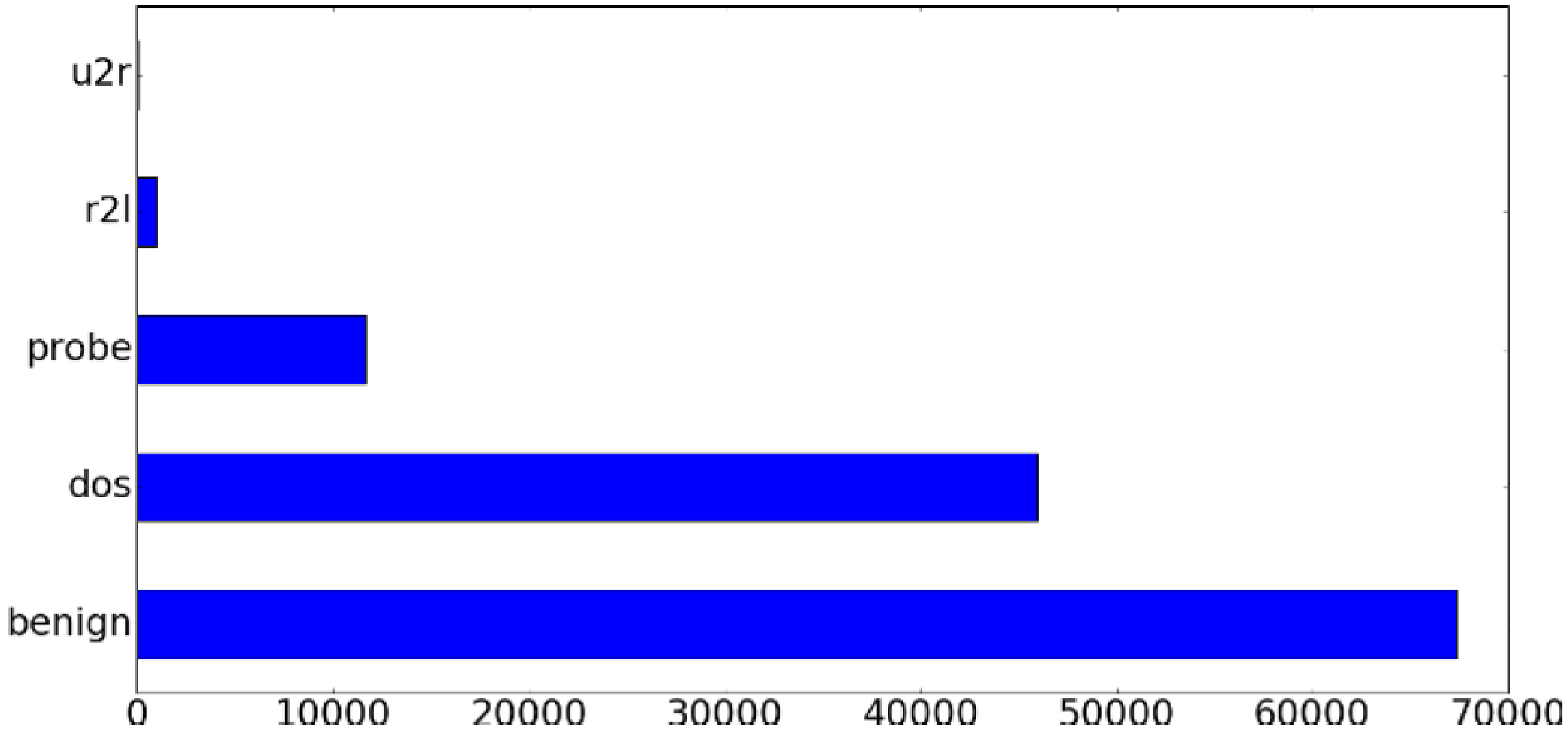


Figure 5-9. Training data class distribution (five-category breakdown)

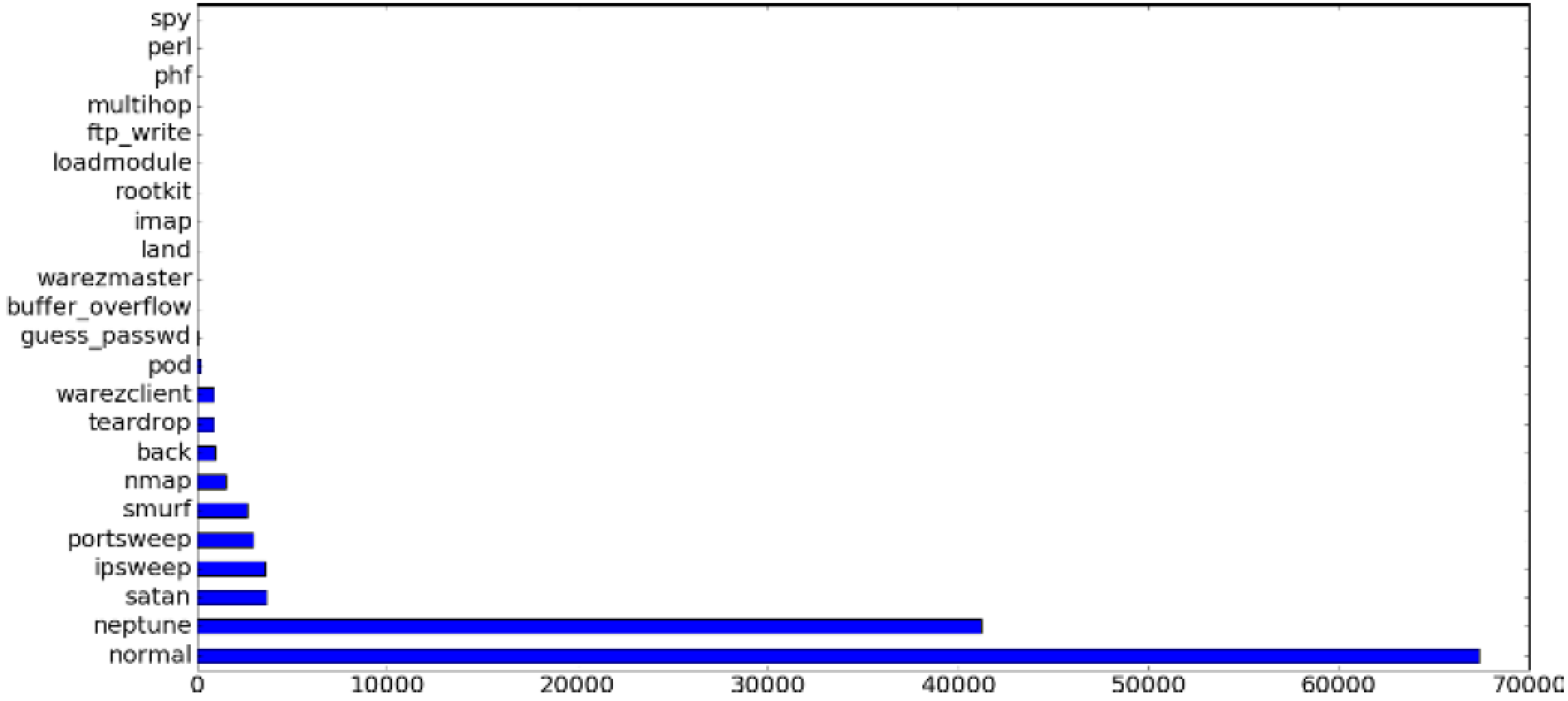
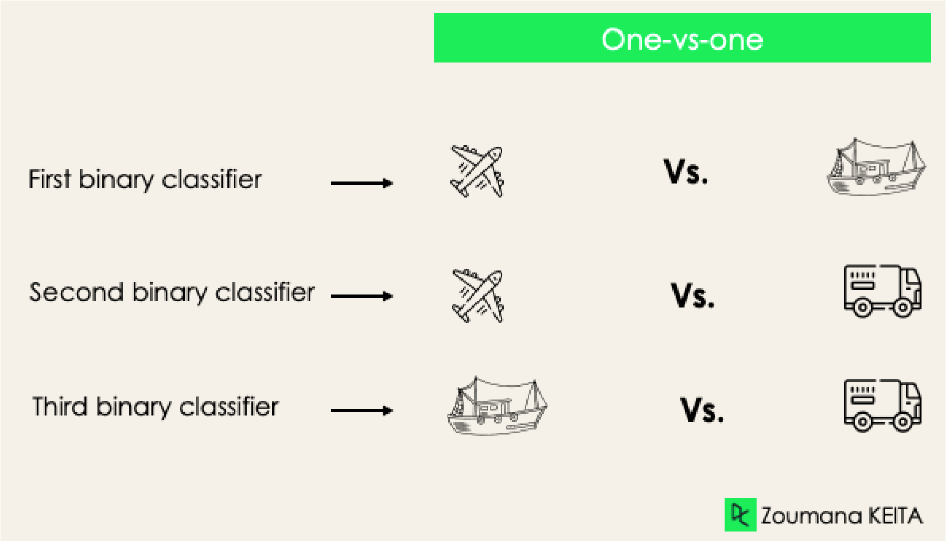
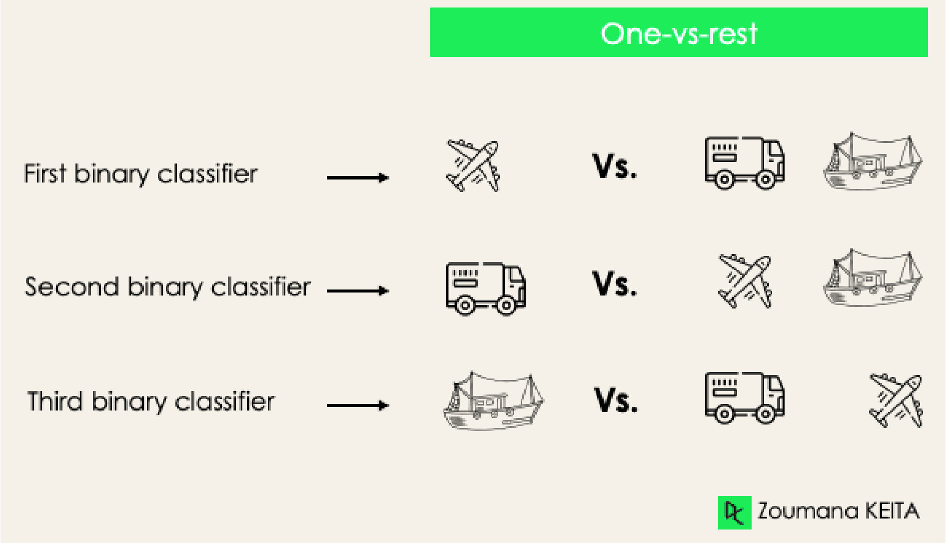


Figure 5-10. Training data class distribution (22-category breakdown)

Classification

- To review, we have a five-class classification problem in which each sample belongs to one of the following classes: benign, u2r, r2l, dos, probe.
- There are many different classification algorithms suitable for a problem like this, and many different ways to approach the problem of multiclass classification.
- Essentially, a multiclass classification problem can be split into multiple binary classification problems.
    - A strategy known as *one-versus-all*, also called the binary relevance method, fits one classifier per class, with data belonging to the class fitted against the aggregate of all other classes.
    - Another less commonly used strategy is *one-versus-one*, in which there are n_classes * (n_classes – 1) / 2 classifiers constructed, one for each unique pair of classes

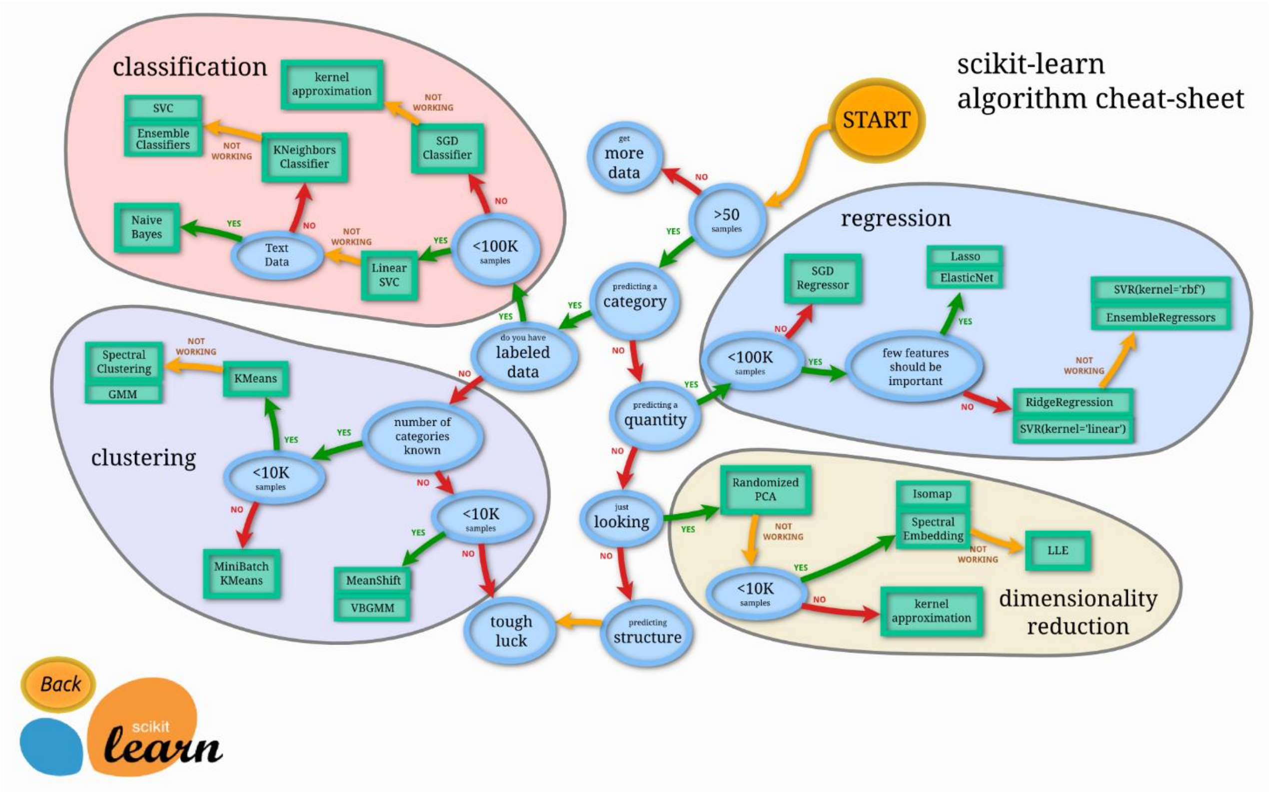


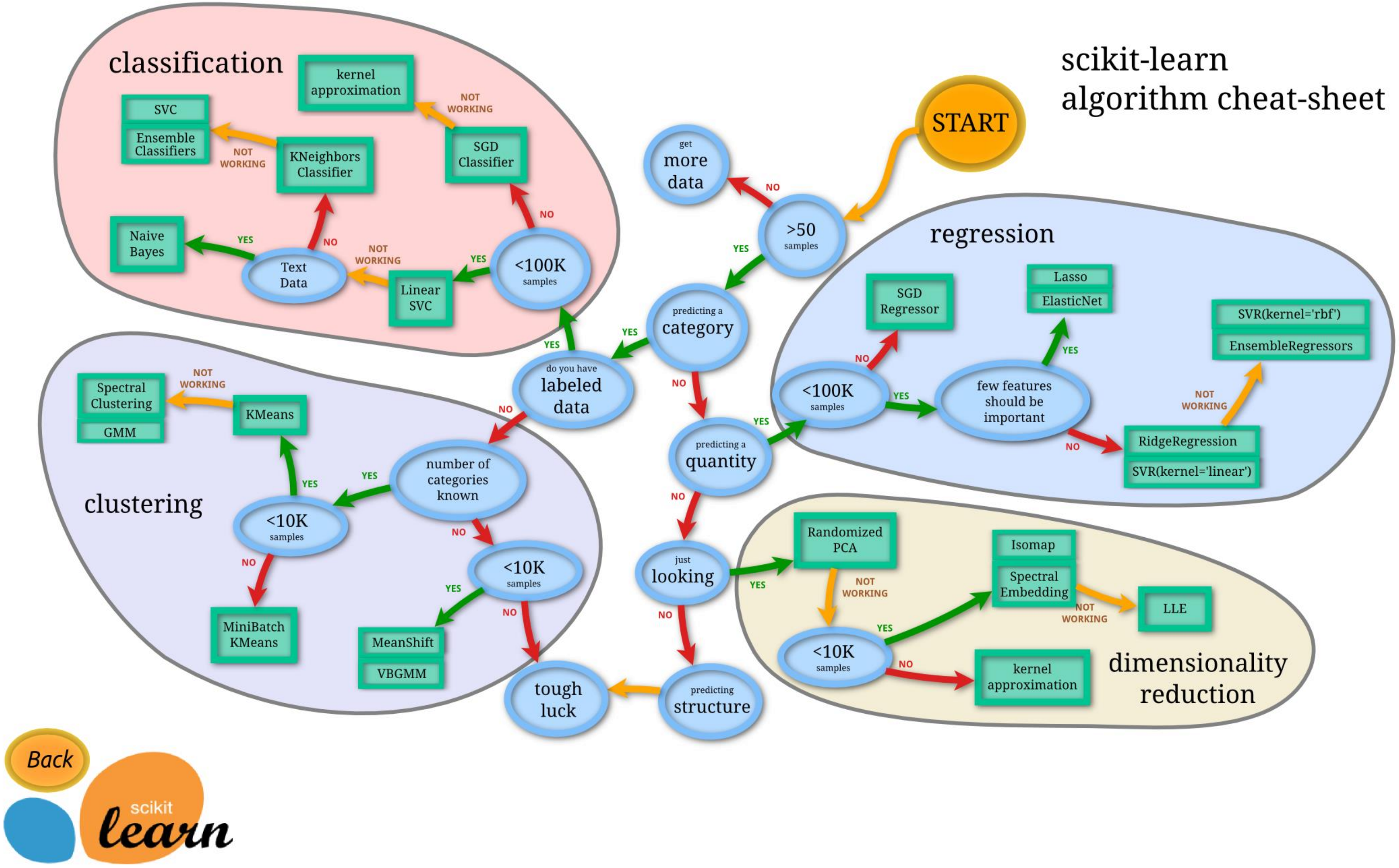


https://www.datacamp.com/blog/classification-machine-learning

Classification

- In general, here are some questions you should ask yourself when faced with machine learning algorithm selection:
- What is the size of your training set?
- Are you predicting a sample's category or a quantitative value?
- Do you have labeled data? How much labeled data do you have?
- Do you know the number of result categories?
- How much time and resources do you have to train the model?
- How much time and resources do you have to make predictions?




scikit-learn
algorithm cheat-sheet
START
>50 samples
get more data
NO
YES
predicting a category
do you have labeled data
predicting a quantity
just looking
predicting structure
tough luck
classification
<100K samples
Linear SVC
Text Data
Naive Bayes
KNeighbors Classifier
SVC
Ensemble Classifiers
SGD Classifier
kernel approximation
NOT WORKING
clustering
number of categories known
<10K samples
KMeans
Spectral Clustering
GMM
MiniBatch KMeans
MeanShift
VBGMM
regression
SGD Regressor
few features should be important
Lasso
ElasticNet
RidgeRegression
SVR(kernel='linear')
SVR(kernel='rbf')
EnsembleRegressors
dimensionality reduction
Randomized PCA
Isomap
Spectral Embedding
LLE
kernel approximation
Back
scikit learn

Classification

- Given that we have access to roughly 126,000 labeled training samples, supervised training methods seem like a good place to begin
- Decision trees or random forests are good places to begin
  - They are invariant to scaling of the data (preprocessing) and
  - They are relatively robust to uninformative features, and hence usually give good training performance

Classification

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix, zero_one_loss

classifier = DecisionTreeClassifier(random_state=0)
classifier.fit(train_x, train_Y)
```

```
pred_y = classifier.predict(test_x)

results = confusion_matrix(test_Y, pred_y)
error = zero_one_loss(test_Y, pred_y)

> # Confusion matrix:
[[9357   59  291    3    1]
 [1467 6071   98    0    0]
 [ 696  214 1511    0    0]
 [2325    4   16  219   12]
 [ 176    0    2    7   15]]

> # Error:
0.238245209368
```

- 76.2% classification accuracy (1 –error rate) in a five-class classification problem is not too shabby. However, this number is quite meaningless without considering the distribution of the test set

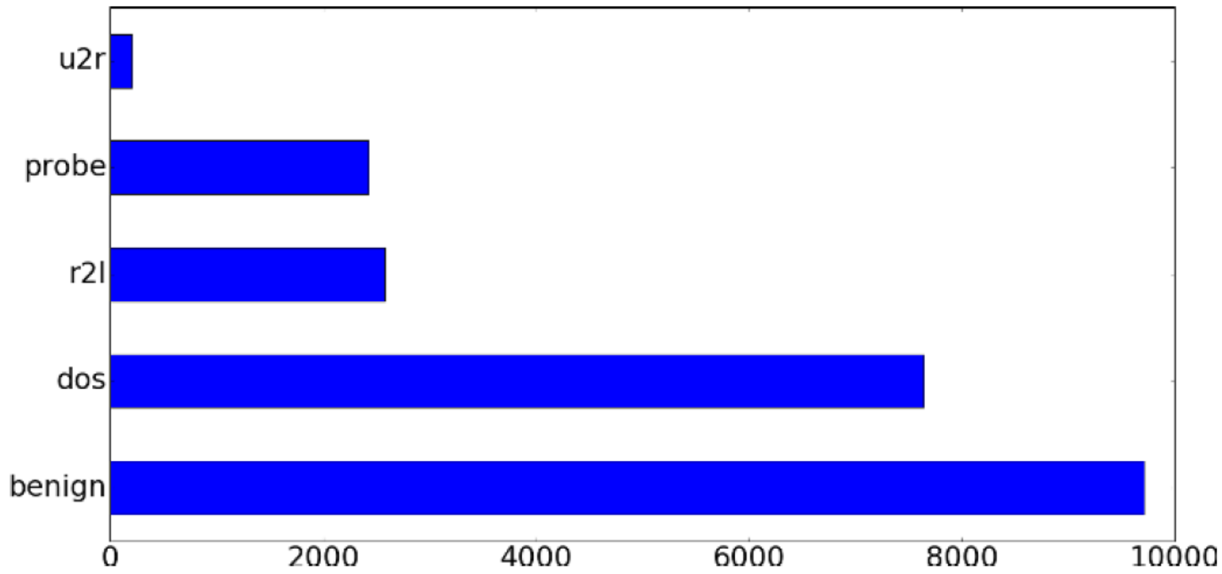


Figure 5-11. Test data class distribution (five-category breakdown)

Class imbalance

- Undersampling
- Oversampling

```
> print(pd.Series(train_Y).value_counts())

    > # Original training data class distribution:
      benign     67343
      dos        45927
      probe      11656
      r2l          995
      u2r           52

from imblearn.over_sampling import SMOTE

# Apply SMOTE oversampling to the training data
sm = SMOTE(ratio='auto', random_state=0)
train_x_sm, train_Y_sm = sm.fit_sample(train_x, train_Y)
print(pd.Series(train_Y_sm).value_counts())

    > # Training data class distribution after first SMOTE:
      benign     67343
      dos        67343
      probe      67343
      u2r        67343
      r2l        67343
```

End