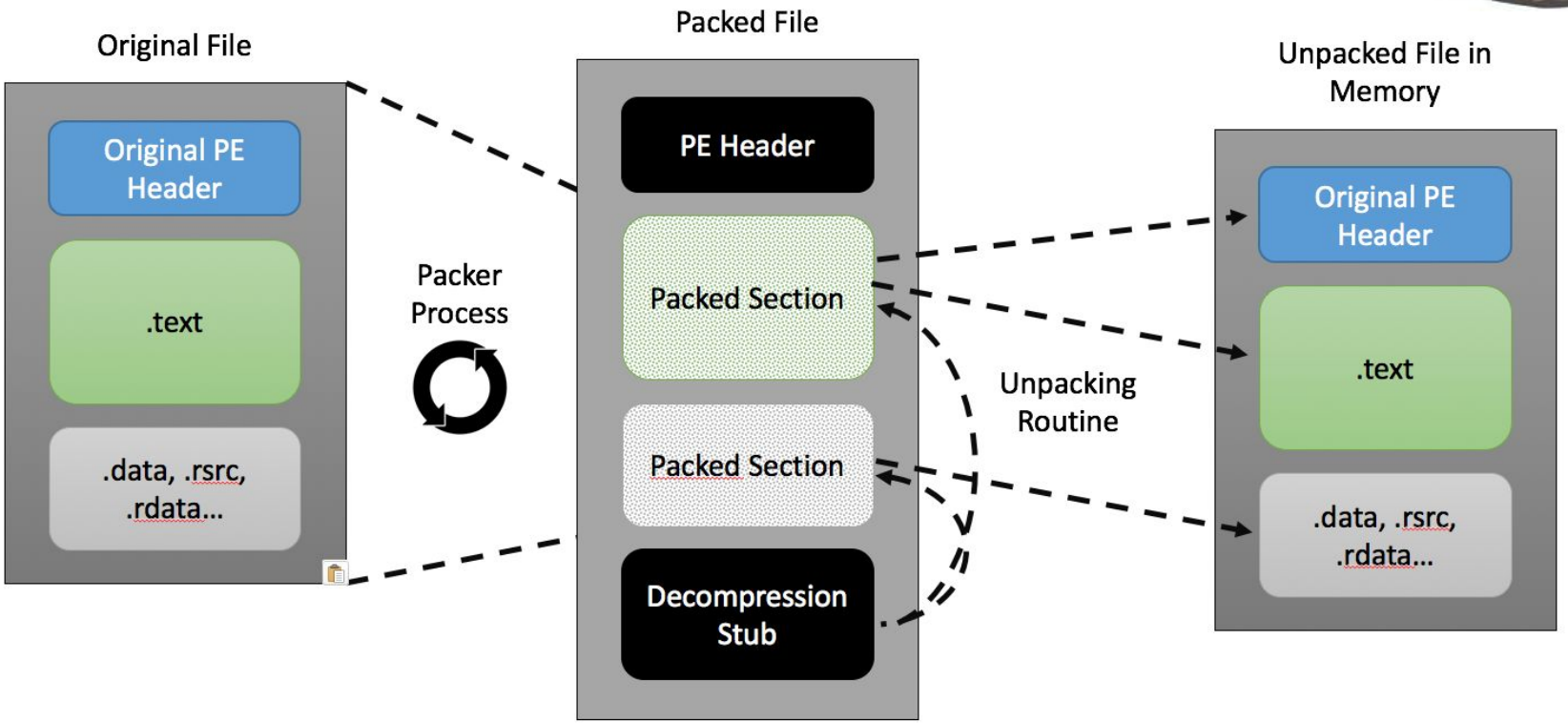
Detecting Packed Executable Files

Projekt za kolegij Strojno učenje

Ivan Čeh
Sebastijan Horvat
PMF, Zagreb, 27. lipnja 2018.

O problemu

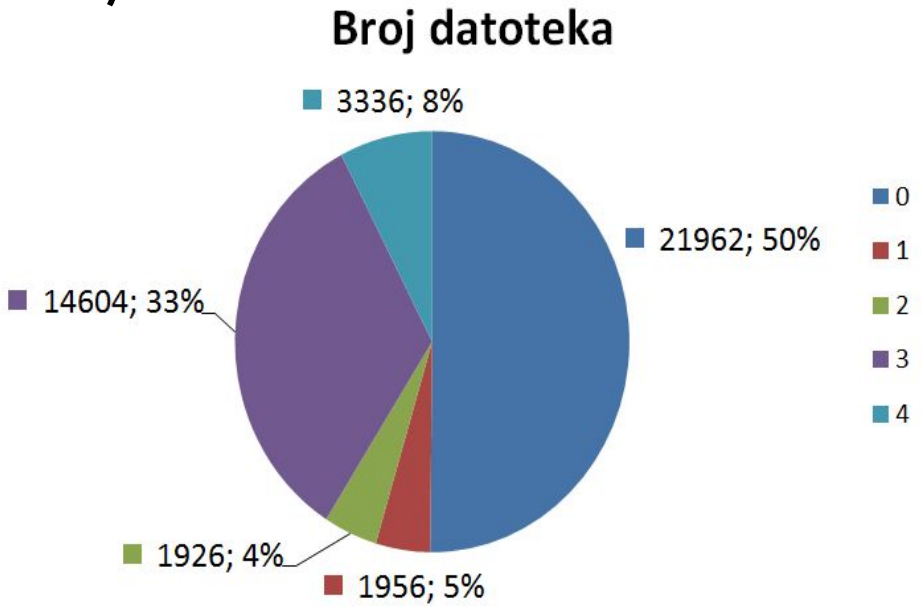

Malware


Original File
Original PE Header
.text
.data, .rsrc, .rdata...
Packer Process
Packed File
PE Header
Packed Section
Packed Section
Decompression Stub
Unpacking Routine
Unpacked File in Memory
Original PE Header
.text
.data, .rsrc, .rdata...

Skup podataka

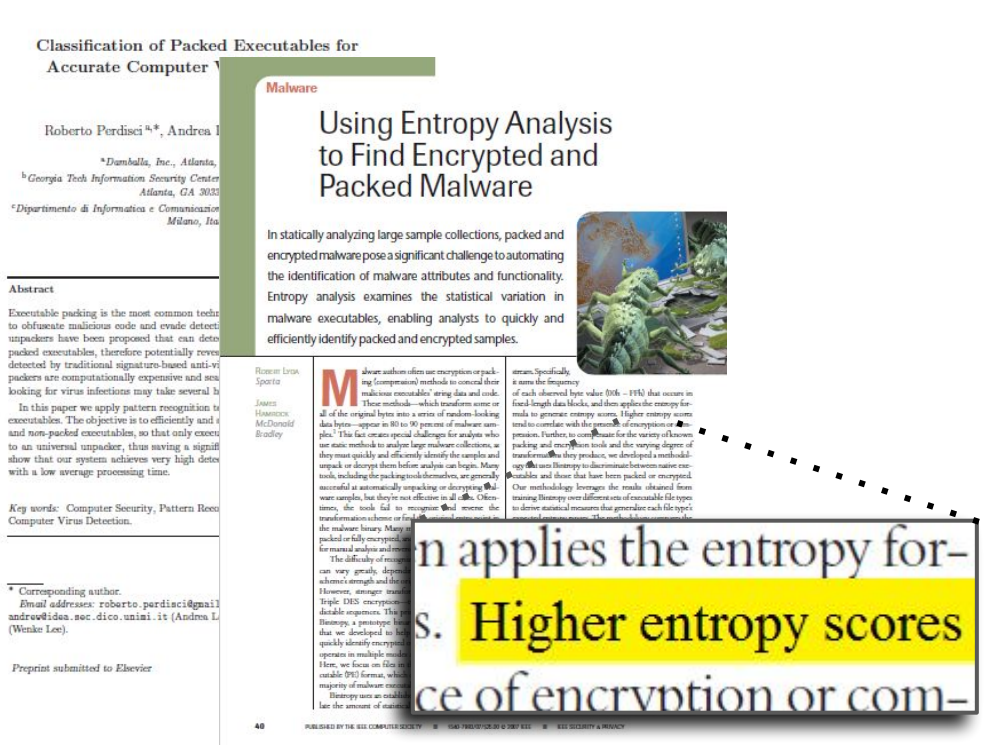
- 43784 izvršivih datoteka i pripadnih izvještaja (označenih)
- u sklopu Mozgala, još 6928 (neoznačenih)

Oznaka	Nepakirano ili vrsta pakiranja
0	nije pakirana
1	pakirana - *Overlay*
2	pakirana - *Crypter*
3	pakirana - *Compressing packer*
4	pakirana - *Protector*



Kako smo tražili značajke?

Literatura



Classification of Packed Executables for Accurate Computer

Roberto Perdisci [a,*], Andrea L

[a] Damballa, Inc., Atlanta,
[b] Georgia Tech Information Security Center, Atlanta, GA 3033
[c] Dipartimento di Informatica e Comunicazion, Milano, Ita

Abstract

Executable packing is the most common techn to obfuscate malicious code and evade detect unpackers have been proposed that can dete packed executables, therefore potentially reves detected by traditional signature-based anti-vi packers are computationally expensive and ses looking for virus infections may take several h

In this paper we apply pattern recognition t executables. The objective is to efficiently and and non-packed executables, so that only exec to an universal unpacker, thus saving a signif show that our system achieves very high dete with a low average processing time.

Key words: Computer Security, Pattern Reco Computer Virus Detection.

* Corresponding author.
Email addresses: roberto.perdisci@gmail andrew@idea.sec.dico.unimi.it (Andrea L (Wenke Lee).

Preprint submitted to Elsevier

Malware

Using Entropy Analysis to Find Encrypted and Packed Malware

In statically analyzing large sample collections, packed and encrypted malware pose a significant challenge to automating the identification of malware attributes and functionality. Entropy analysis examines the statistical variation in malware executables, enabling analysts to quickly and efficiently identify packed and encrypted samples.

Robert Lyda
Sparta

James Hamrock
McDonald Bradley

Malware authors often use encryption or packing (compression) methods to conceal their malicious executables' string data and code. These methods—which transform some or all of the original bytes into a series of random-looking data bytes—appear in 80 to 90 percent of malware samples.[1] This fact creates special challenges for analysts who use static methods to analyze large malware collections, as they must quickly and efficiently identify the samples and unpack or decrypt them before analysis can begin. Many tools, including the packing tools themselves, are generally successful at automatically unpacking or decrypting malware samples, but they're not effective in all cases. Oftentimes, the tools fail to recognize or fully reverse the transformation scheme or find the original entry point in the malware binary. Many

stream. Specifically, it sums the frequency of each observed byte value (0x0h – FFh) that occurs in fixed-length data blocks, and then applies the entropy formula to generate entropy scores. Higher entropy scores tend to correlate with the presence of encryption or compression. Further, to compensate for the variety of known packing and encryption tools and the varying degree of transformation they produce, we developed a methodology that uses Bintropy to discriminate between native executables and those that have been packed or encrypted. Our methodology leverages the results obtained from training Bintropy over different sets of executable file types to derive statistical measures that generalize each file type's

n applies the entropy for-
s. Higher entropy scores
ce of encryption or com-

40 PUBLISHED BY THE IEEE COMPUTER SOCIETY IEEE SECURITY & PRIVACY

Proučavanje izvještaja


```
                "address":"0x00000000",
                "size":"0x00000000"
            },
            {
                "address":"0x00000000",
                "size":"0x00000000"
            }]
        }
    },
    "sections":{
        "section_list": [
        {
            "name":".text",
            "flags":{
                "flag_list": [
                "IMAGE_SCN_CNT_CODE",
                "IMAGE_SCN_MEM_EXECUTE",
                "IMAGE_SCN_MEM_READ"]
            },
            "size":"0x00001800",
            "address":"0x00001000",
            "offset":"0x00000400",
            "entropy":"5.8876270060638998"
        },
        {
            "name":".rdata",
            "flags":{
                "flag_list":
                "IMAGE_SCN_CN
                "IMAGE_SCN_ME
            },
            "size":"0x000014
            "address":"0x000
            "offset":"0x0000
            "entropy":"5.148
        },
        {
            "name":".data",
            "flags":{
                "flag_list":
                "IMAGE_SCN_CNT_INITIALIZED_DATA",
                "IMAGE_SCN_MEM_READ",
                "IMAGE_SCN_MEM_WRITE"]
            },
```



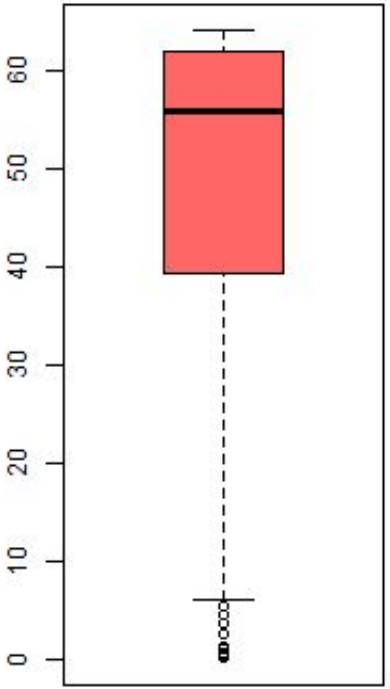
```
            },
            "size":"0x00001800",
            "address":"0x00001000",
            "offset":"0x00000400",
            "entropy":"5.8876270060638998"

            "name":".rdata",
```

Težinska sredina kvadrata entropija sekcija

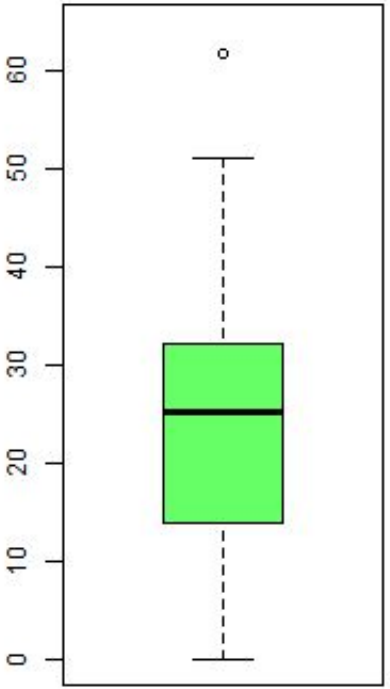
	veličina	entropija
Section 1	0x1800	5.8876
Section 2	0x1400	5.1488
Section 3	0x200	0.4524
Section 4	0x200	3.7829
Section 5	0x200	0.7613

27.8467

Pakirane



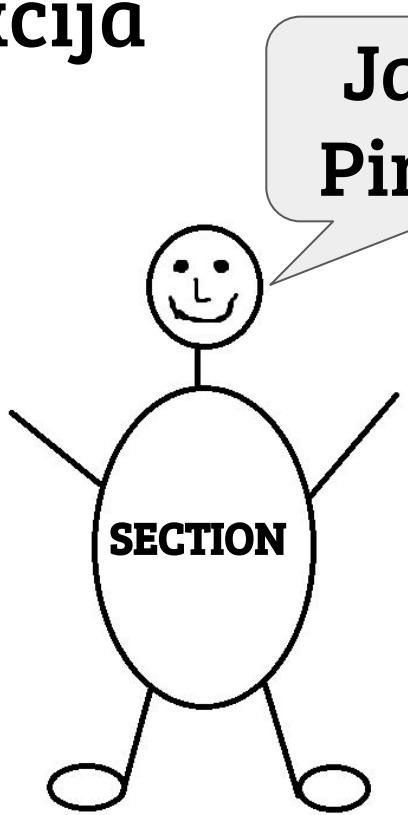
Nepakirane

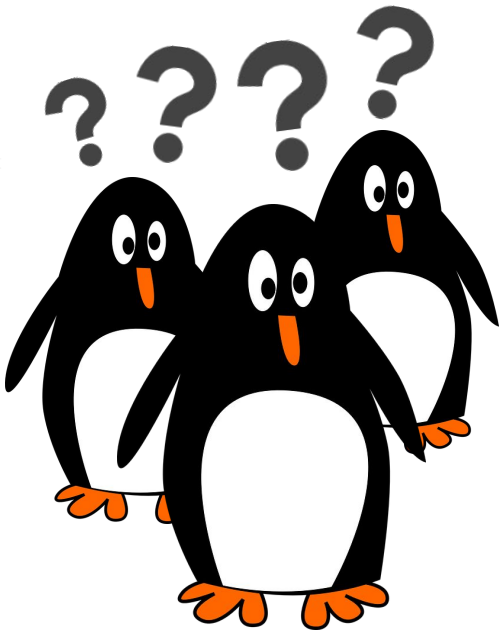


Nestandardna imena sekcija

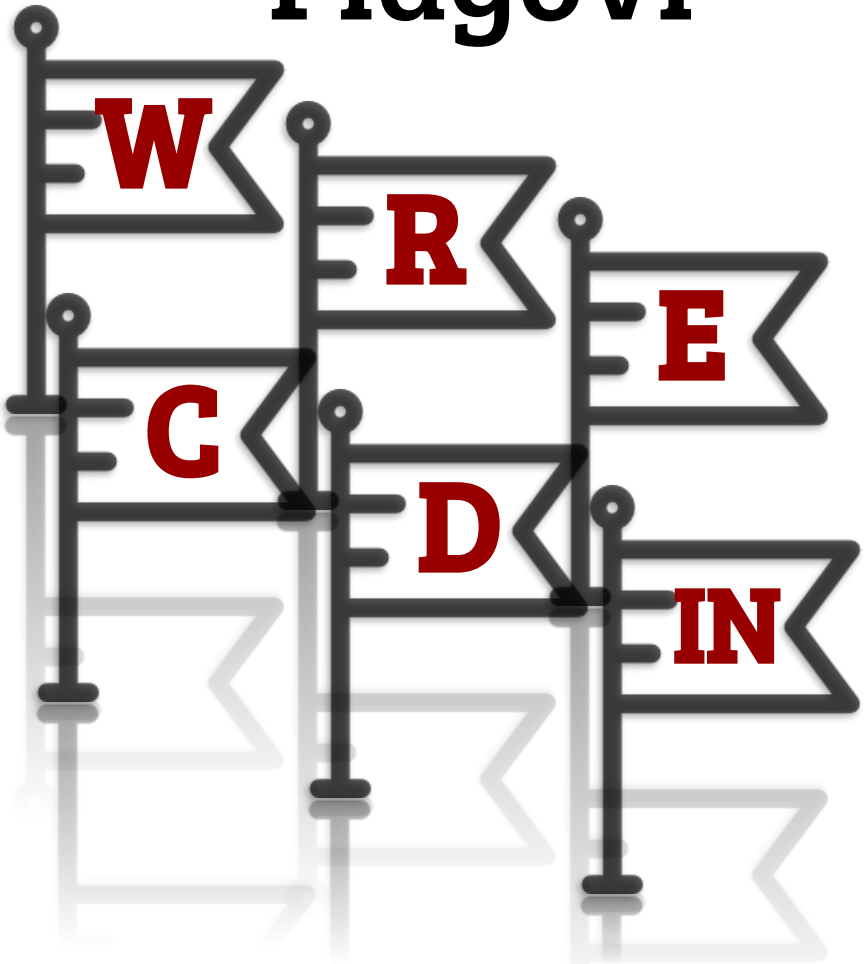
.text	.reloc
.data	.sdata
.rdata	.srdata
.idata	.pdata
.edata	.debug$S
.rsrc	.debug$T
.bss	.debug$P
.crt	.drectve
.tls	.didat

\+ .imports

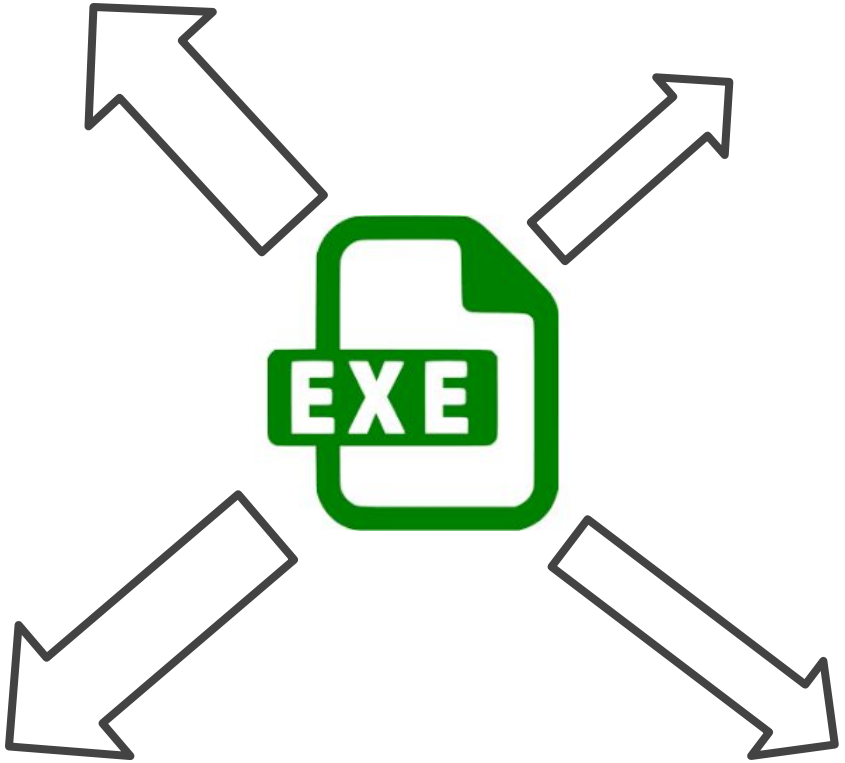




Flagovi



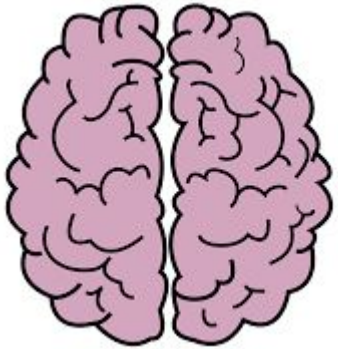
API-ji



Složenije značajke

sizeOfInitializedData
(suma po svim Entrijima)

sizeOfUninitializedData
(suma po svim Entrijima)



$$\log\left(\frac{1 + initalized}{1 + uninitialzed}\right)$$

---

entropy (section#1)

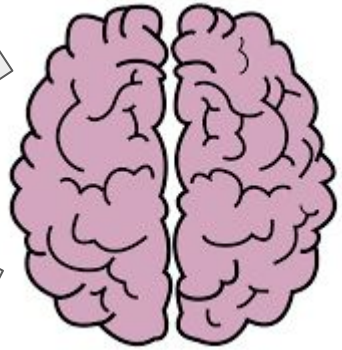
⋮

entropy (section#n)

size (section#1)

⋮

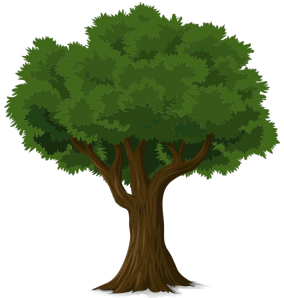
size (section#n)



$$\max_{s \text{ je sekcija}} \left(\frac{\text{size}(s)}{\text{entropy}^2(s)}\right)$$

SLUČAJNA ŠUMA

scikit learn

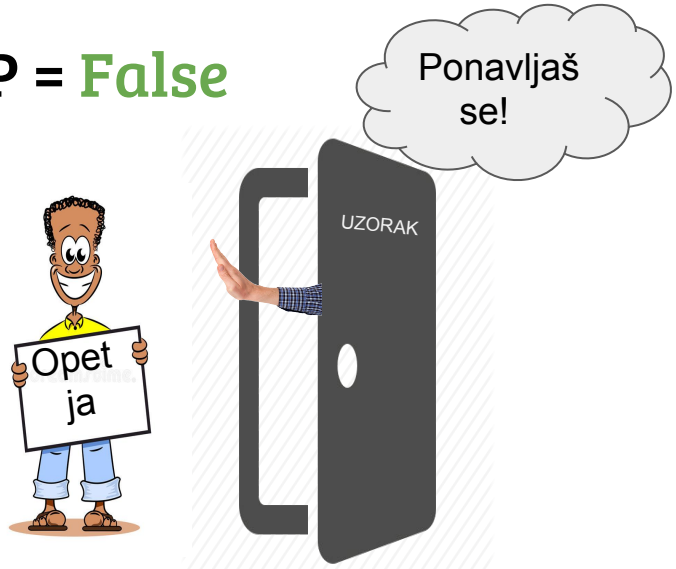


× 800

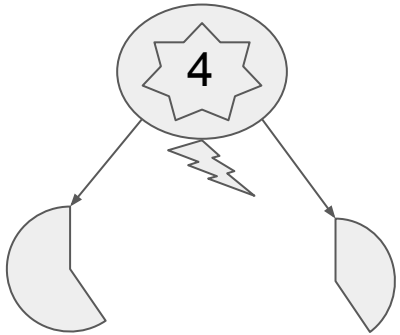
RANDOM STATE

PARAMETRI MODELA

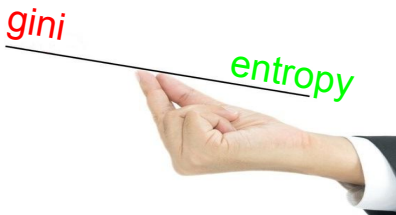
BOOTSTRAP = False



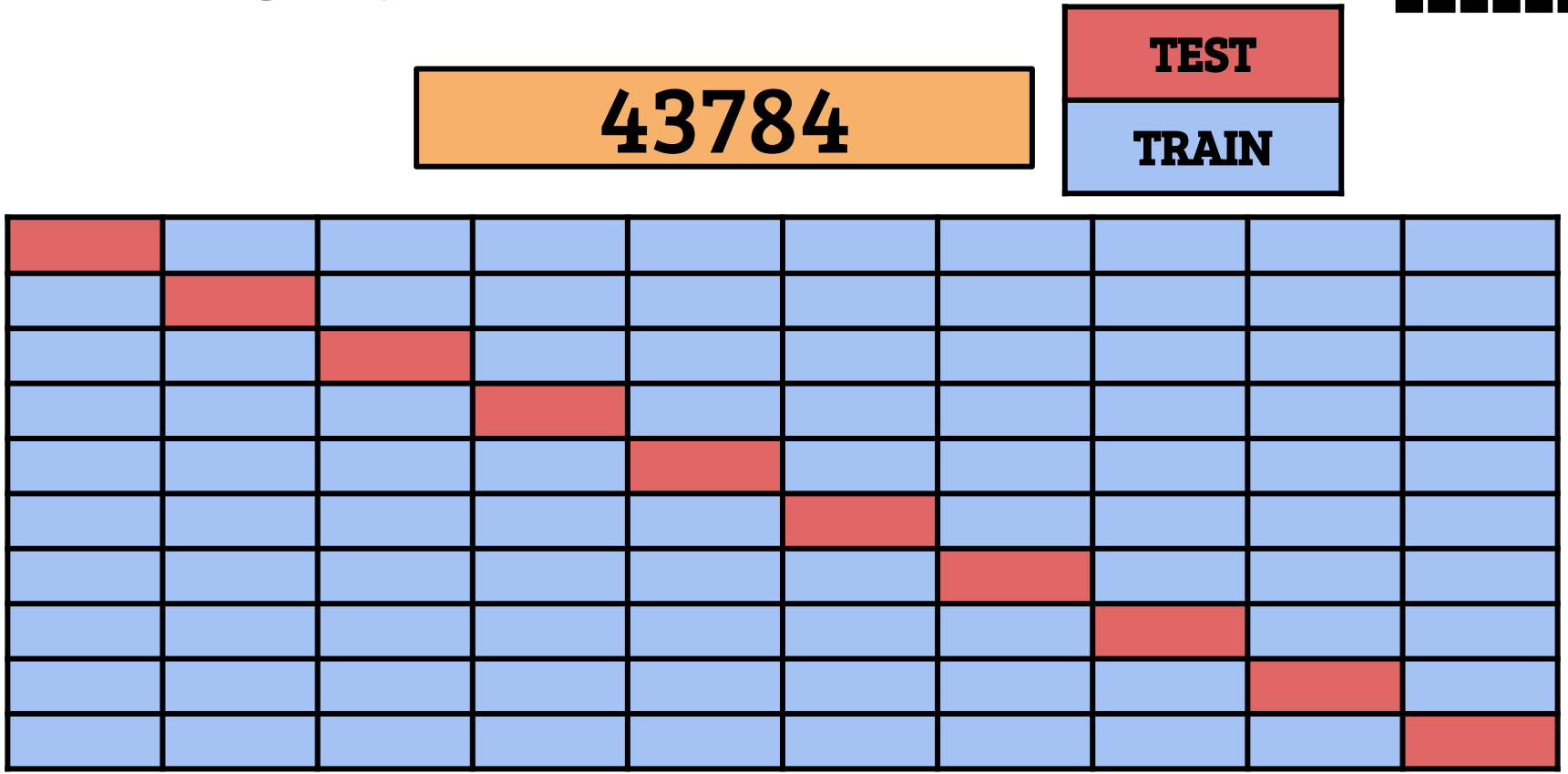
MIN. SAMPLES SPLIT



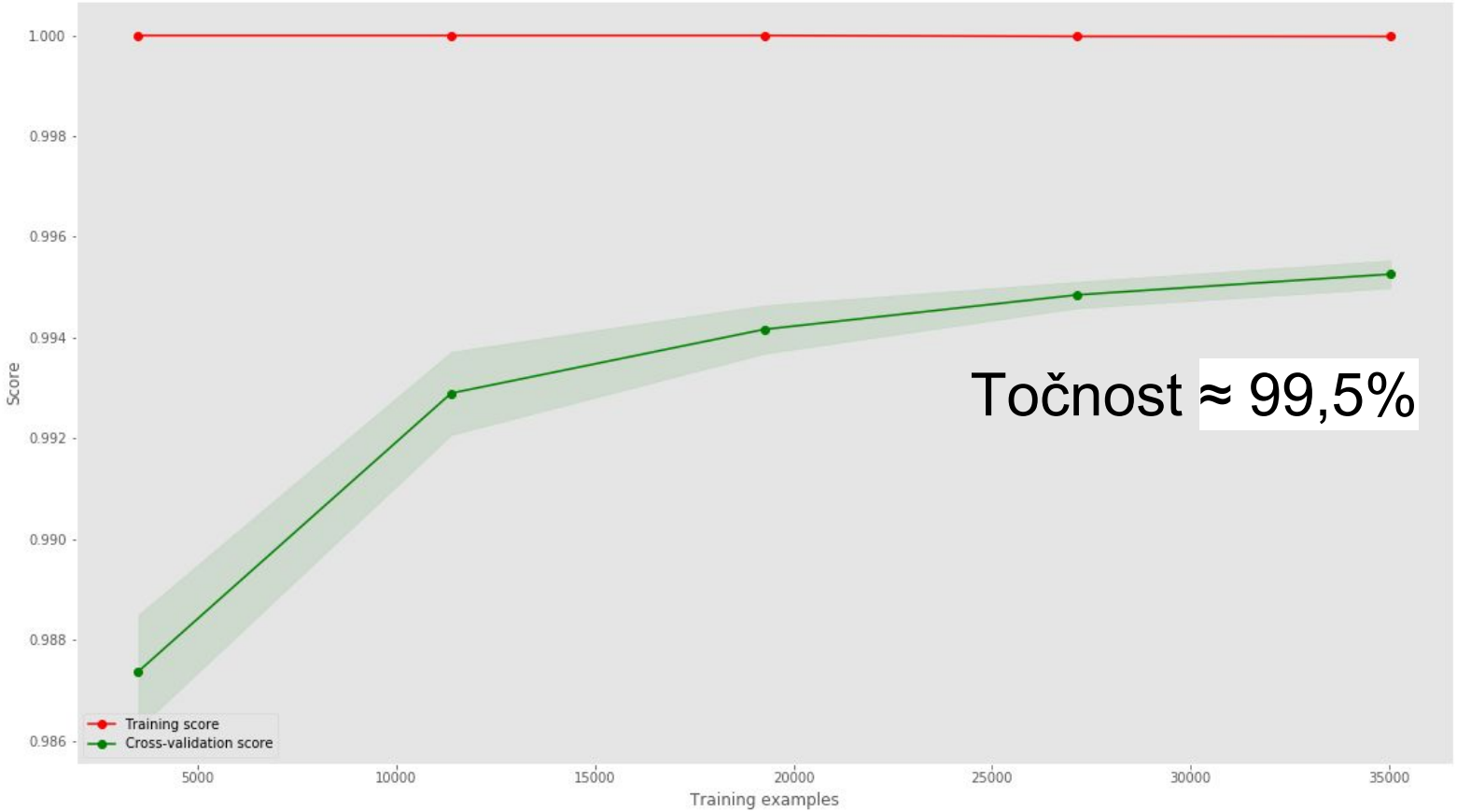
CRITERION



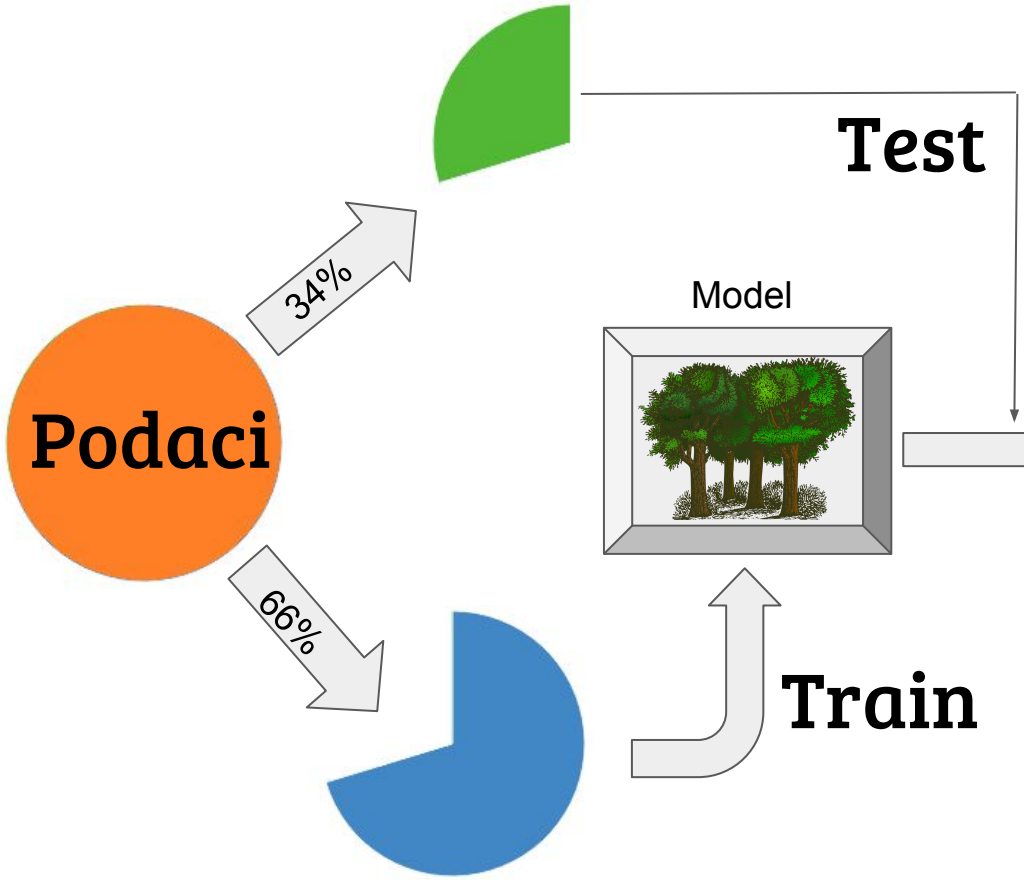
10-fold



KRIVULJA UČENJA

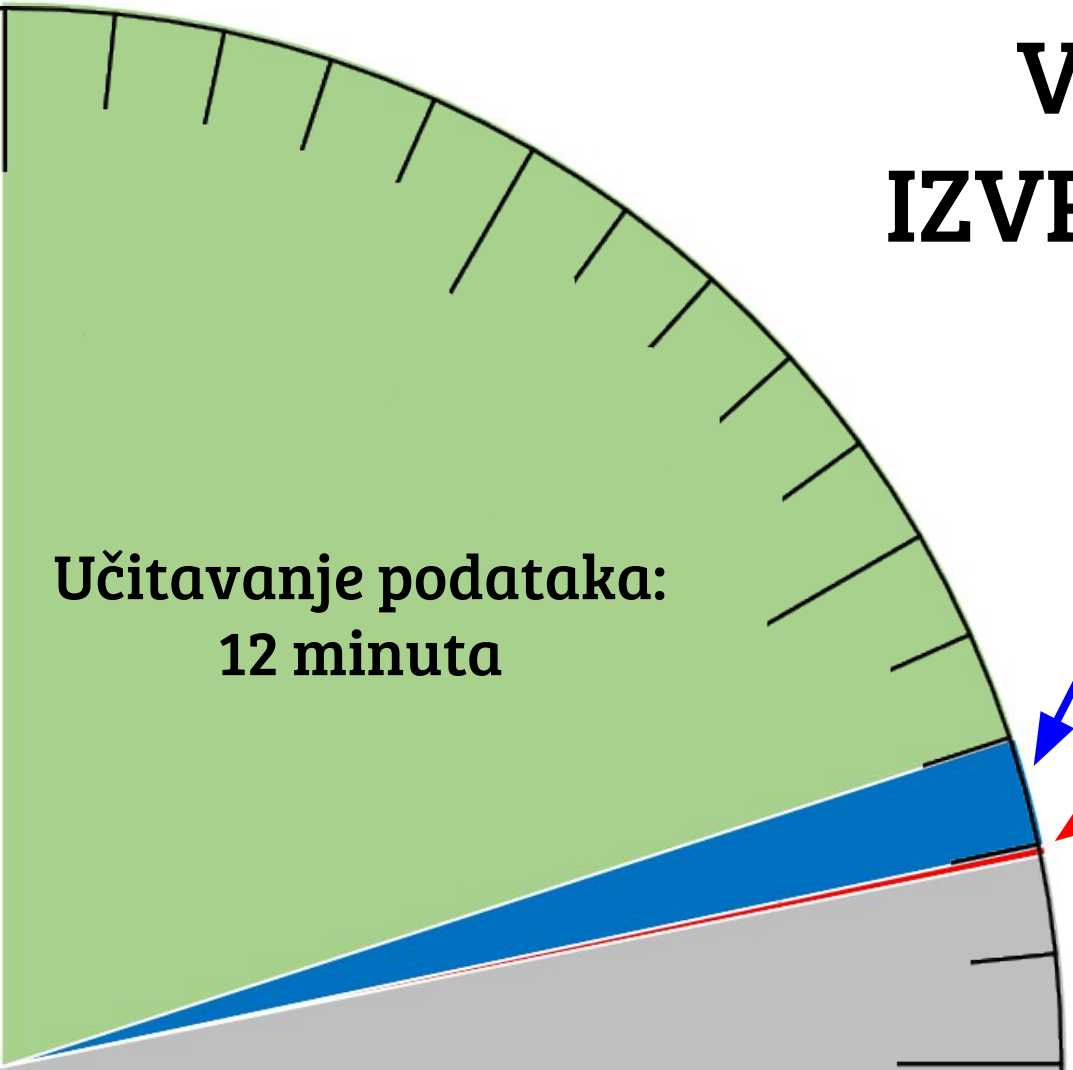


Jedna matrica konfuzije



prediction original	False	True
t0	7371	18
t1	33	640
t2	10	688
t3	5	5013
t4	2	1107

VRIJEME IZVRŠAVANJA



Trening: 1 minuta

Predikcija (14 000 datoteka):
5 sekundi

Windows 8.1, Pentium Dual-Core
2.00 GHz, 32b, 4GB RAM

Rezultati natjecanja Mozgalo 2018.

6928 testnih primjeraka

točnost: 97.97921%

točnost na neviđenim primjerima:
536/556 = 96.40288%

prediction / original	False	True
t0	3399	63
t1	6	479
t2	46	625
t3	3	1147
t4	22	1138

Rezultati natjecanja Mozgalo 2018.

Naziv tima	Točnost (%)	FP+FN
Laganica ❸	98.239	122
NewTeam (mi) ❷	97.979	140
Analysis Paralysis	97.546	170
GMO Lazo ❶	97.373	182
505	95.901	284

Pitanja?