How to NEUTRALIZE Machine Learning based Anti-Malware Software

JunSeok Seo (boanproject) + **JaeHwan Kim** (Korea Univ)

2017. 7. 12

Who we are

- **Jun-Seok, Seo (nababora)**
  - Vice President of Boanprjoect (start-up)
  - Study for Teaching – Vuln Analysis, IoT, ML, Malware
  - Interested in AI, ML, especially ‘adversarial ML’
  - nababora@naver.com

- **Jae-Hwan, Kim**
  - Researcher, Data Scientist
  - Interested in Machine Learning for data analysis
  - edenkim519@korea.ac.kr

Background

- We live in the data-driven world, everything is data
- We have no choice but to use ‘data’, ‘machine learning’, ‘AI’
- AI uses machine learning as a core engine
- Machine learning is de facto ultimate solution in information security...?!
- Can we fully trust decision made by machines ?

What if ?!


90
MLB 040


ZU 0666',0,0); DROP DATABASE TABLICE;

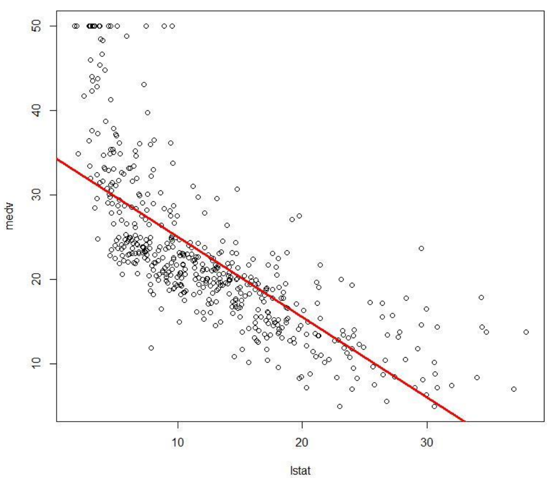

SKYNET
CYBERDYNE SYSTEMS CORPORATION

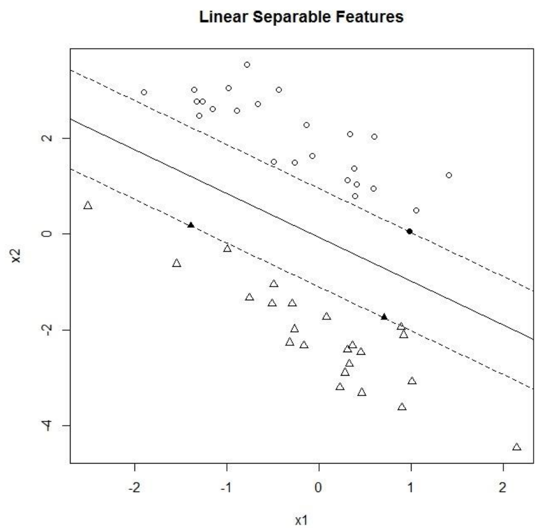
ML in Information Security

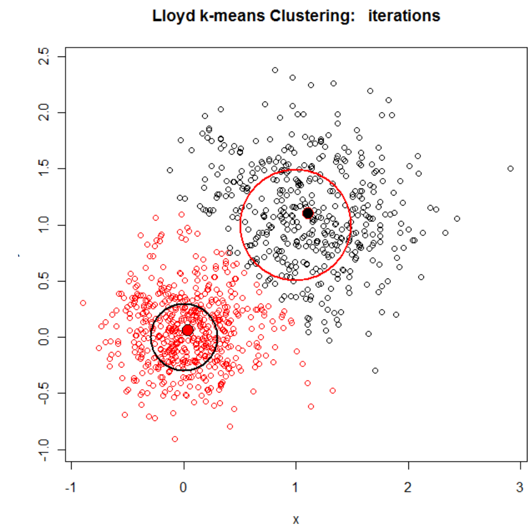
- **Spam Filtering**
  - Based on probabiility of each word in e-mail contents
- **Network Traffic Analysis**
  - Find malicious traffic with anomaly detection
- **Incident Prevention & Response**
  - Find abnormal ‘PATTERN’ in data (system log, traffic, application log, etc)
- **Malware Detection**
  - What I am going to show you today

What is ML

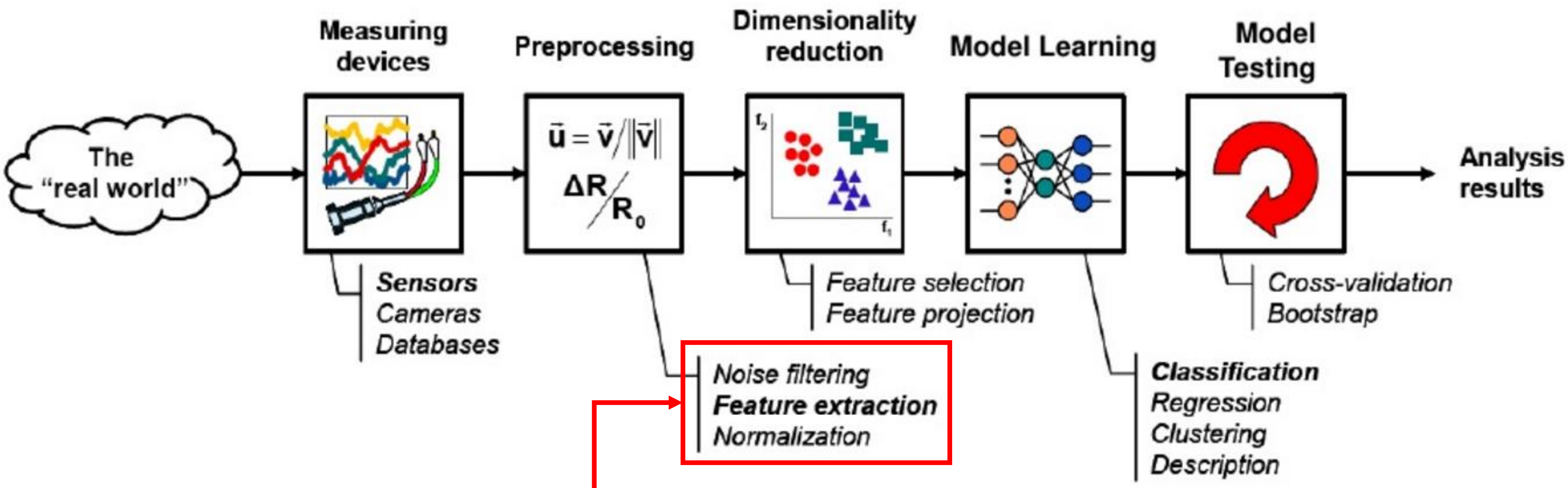
- **Machine Learning is**
  - computers the ability to learn without being explicitly programmed
  - explores the study and construction of algorithms that can learn from and make predictions on data
  - It is just the way of drawing a line (what ? how ? where ?)







ML Process

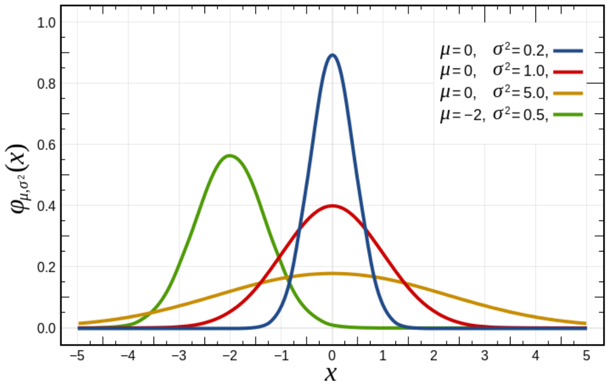


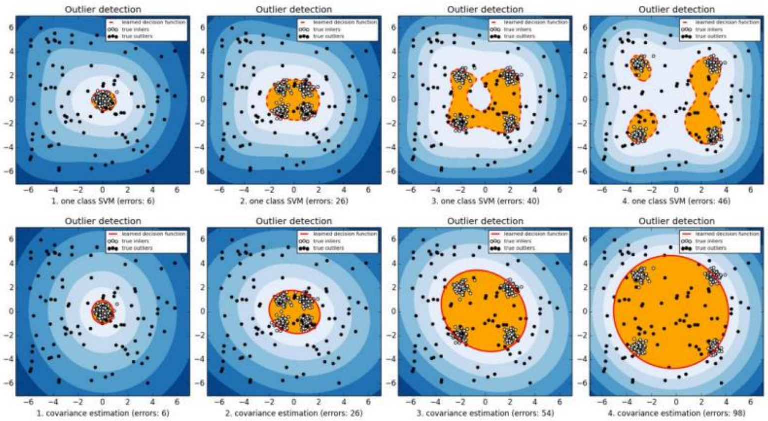
[Gutierrez-Osuna (2011), PRISM]

' FEATURE ' is the key !

So, How to learn?

- **Probability distribution**
- **Correlation Analysis**
- **Euclidean Distance**
- **Entropy**
- **Bayes Theorem**
- **BackPropagation**
- **...**

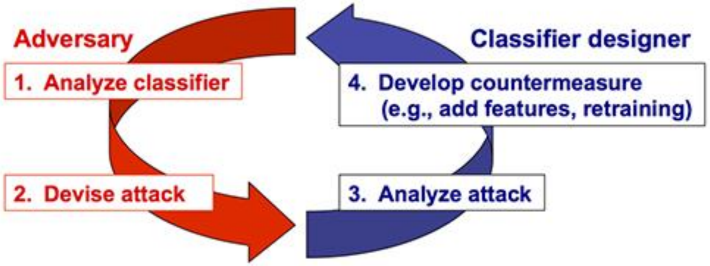


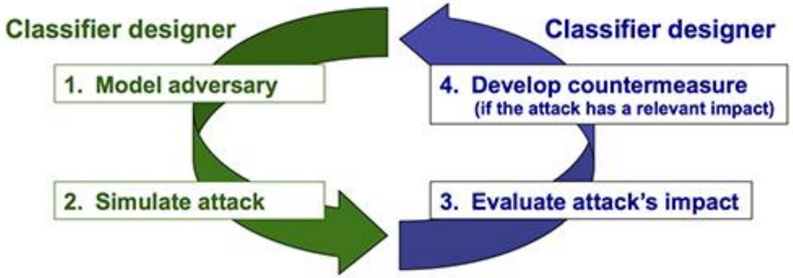


http://www.hongyusu.com/programming/2015/10/10/novelty-detection/

Advarsarial Machine Learning

- **Advarsarial Machine Learning is**
  - research field that lies at the intersection of ML and computer security
  - it aims not only to violate security, but also to compromise the learnability
- **Arms Race Problem**
  - arms race between the adversary and the learner
  - ‘reactive’ (security by obscurity) / ‘proactive’ (security by design)





Adversaral Examples

- **EvadeML:** forces a malicious PDF detector(ML) make wrong predictions
  - **https://githubcom/uvasrg/EvadeML**
- **AdversariaLib:** algorithms focused on sklearn and neural networks
  - http//pralab.diee.unica.it/en/AdversariaLIb
- **Explaining and Harnessing Adversarial Examples**
  - https://pdfs.semanticscholar.org/bee0/44c8e8903fb67523c1f8c105ab4718600cdb.pdf
- **Pwning Deep Learning Systems**
  - https://www.slideshare.net/ClarenceChio/machine-duping-101-pwning-deep-learning-systems

Examples: Spam Filtering

- **Spam Filtering with 'WORD' probability**

$$P(spam|penis, viagra)$$

$$= \frac{P(penis|spam) * P(viagra|spam) * P(spam)}{P(penis) * P(viagra)}$$

$$= \frac{\frac{24}{30} * \frac{20}{30} * \frac{30}{74}}{\frac{25}{74} * \frac{51}{74}} = 0.928$$

- **It's black boxed, but**

https://alexn.org/blog/2012/02/09/howto-build-naive-bayes-classifier.html

Attack Taxonomy

	Causative (online learning)	**Exploratory**
Targeted	Classifier is mis-trained on particular positive samples	Misclassifying a specific subset of positive samples
Indiscriminate	Classifier is mis-trained generally on positive samples	Misclassifying positive samples generally

on training phase

on testing phase

Attack Taxonomy

- **Targeted Exploratory Integrity Attack (TEIA)**

$$\mathcal{A}^* = \arg \max_{\mathcal{A} \in \Phi(\mathcal{A})} FN(y_i, f(x_i)), \quad (x_i, y_i) \in \mathcal{A}$$

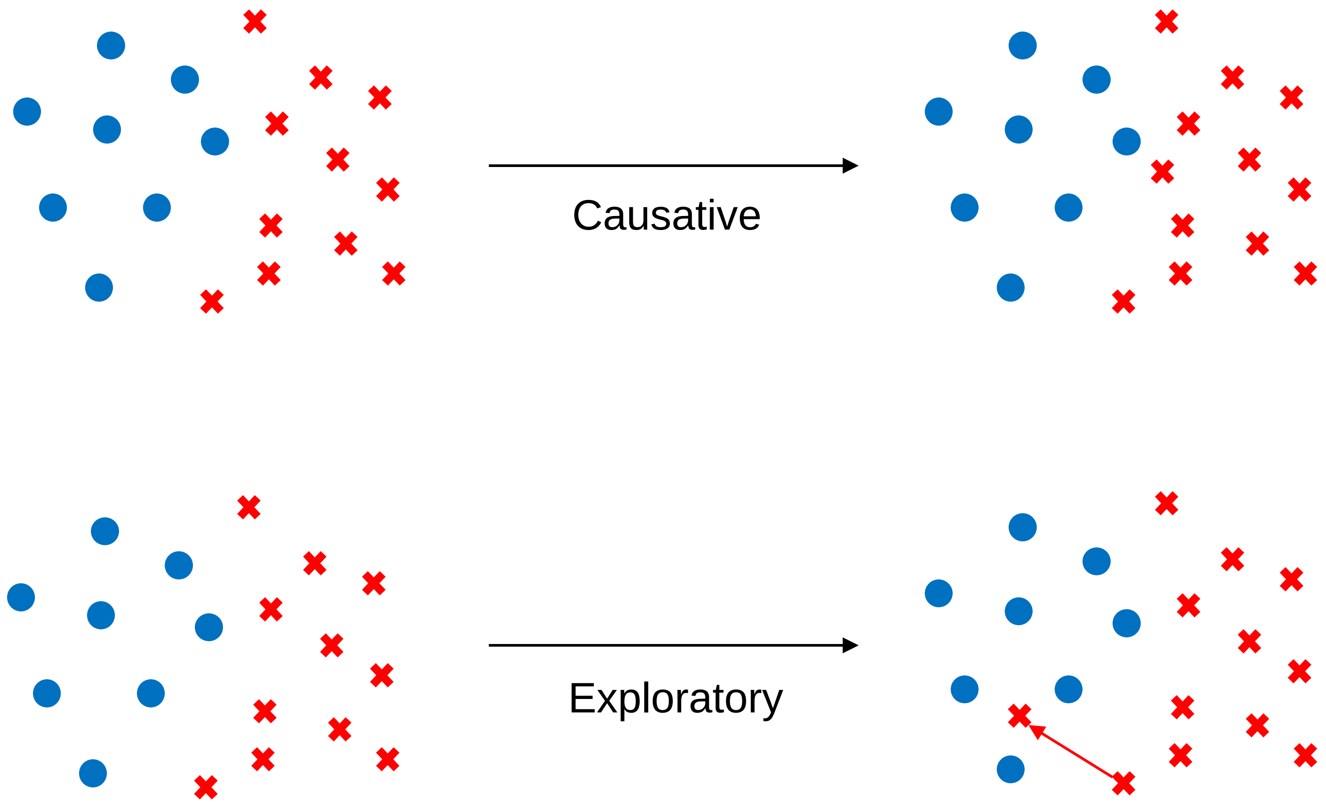
- It's based on the 'Game Theory' - maximize the false negative
- **condition**: 'the number of permitted queries is sufficiently large'
- but, can you understand this formula?

<u>* False Negative - a test result indicates that a condition failed, while it was successful</u>

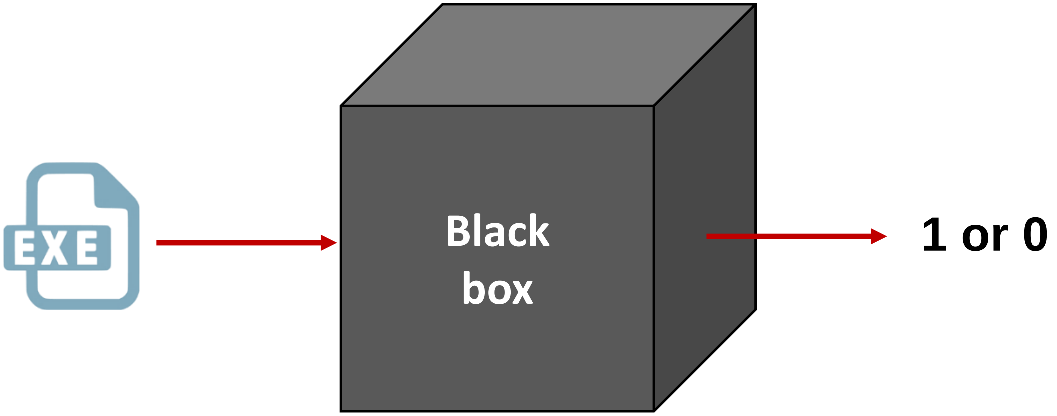
Attack Taxonomy

Intuition, rather than **formula**

Attack Model

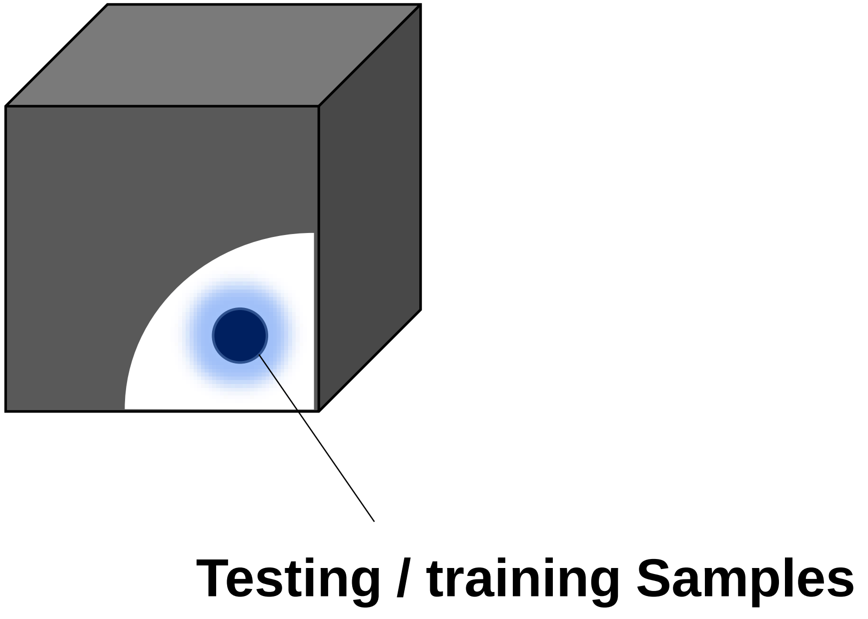


Adversary Knowledge

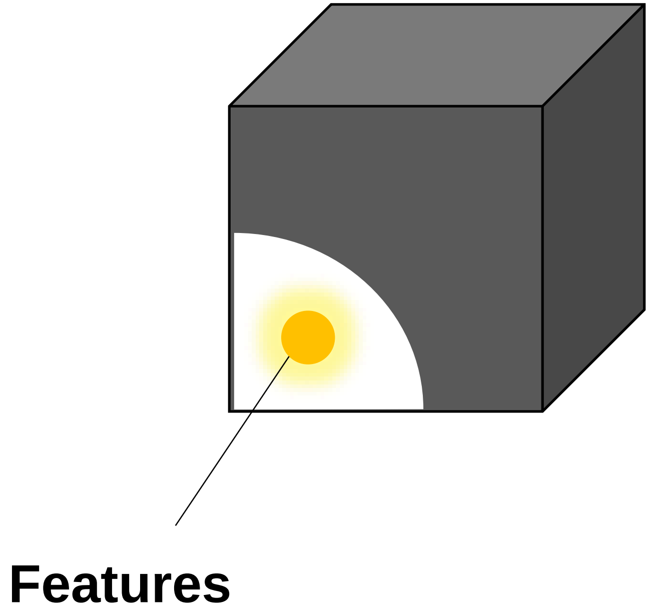


Zero Knowledge = only input and output

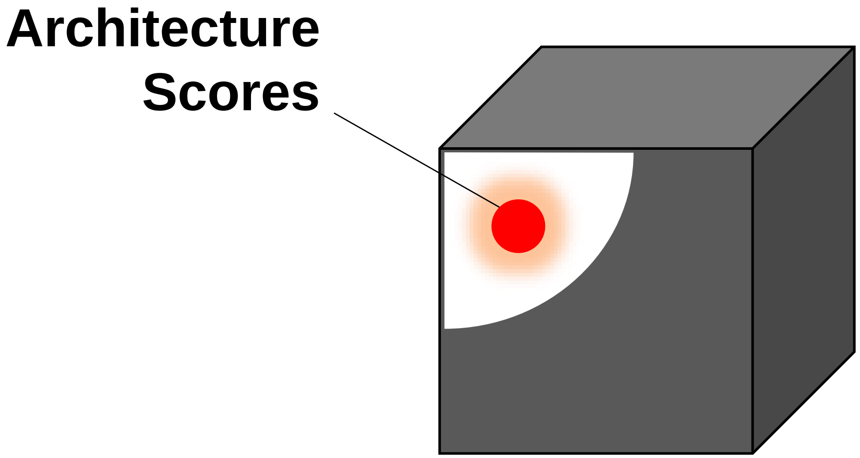
Adversary Knowledge



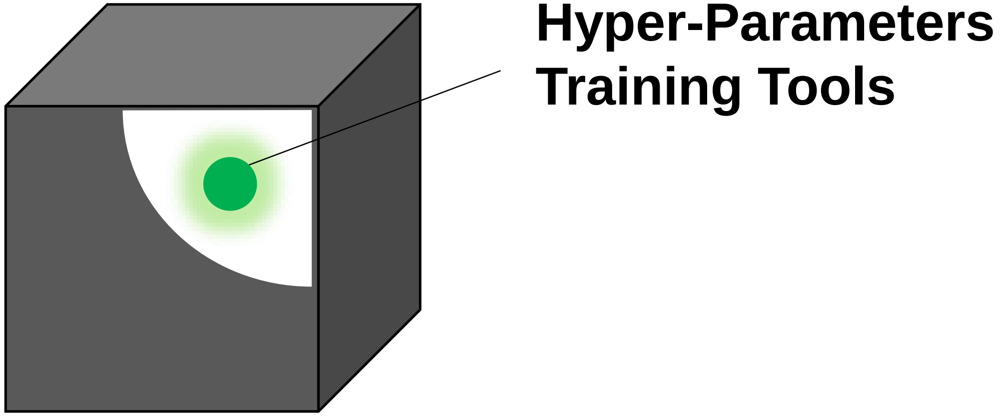
Adversary Knowledge



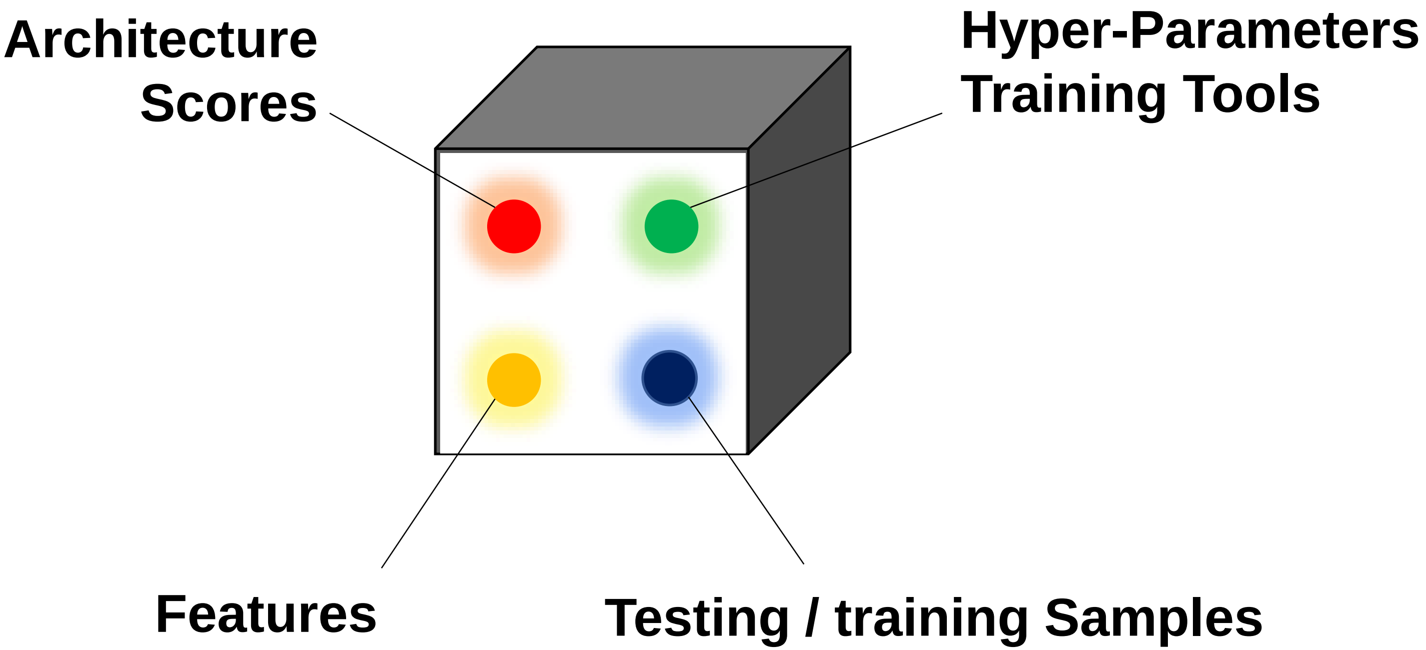
Adversary Knowledge



Adversary Knowledge



Adversary Knowledge

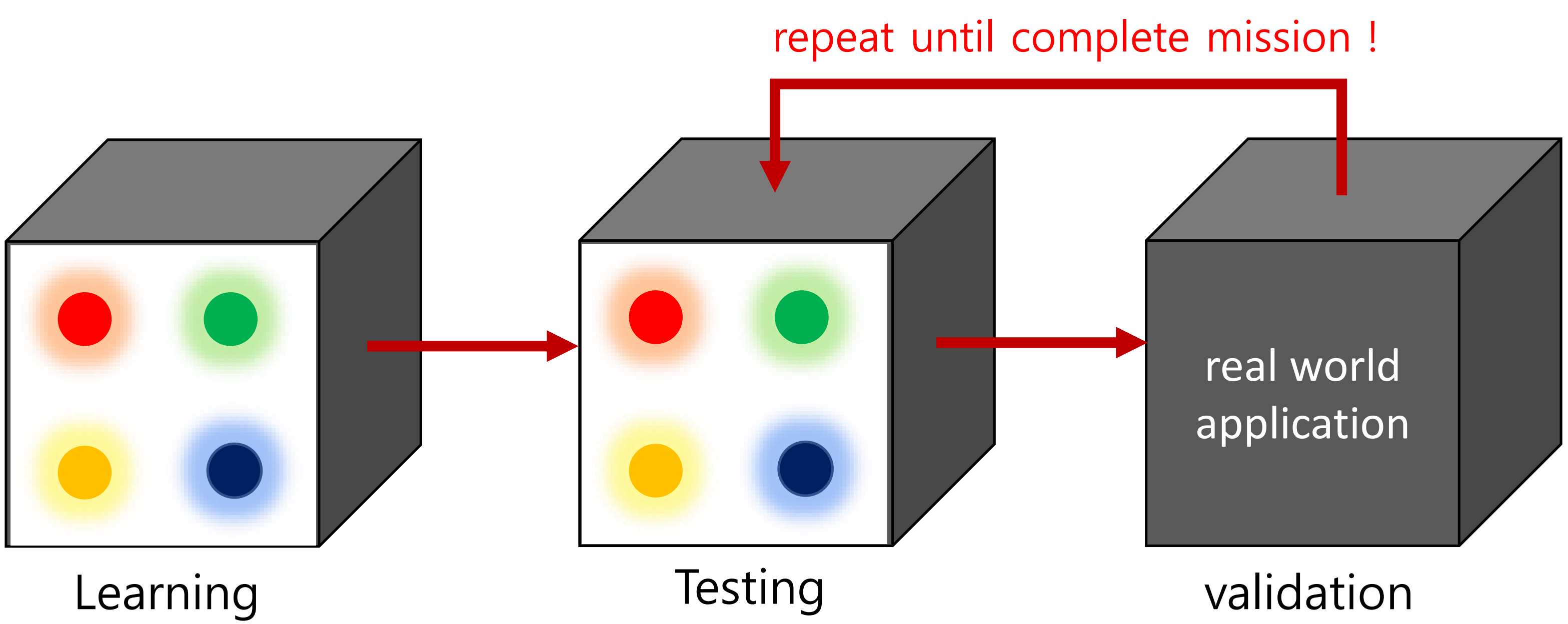


In the real world, none of them are available !

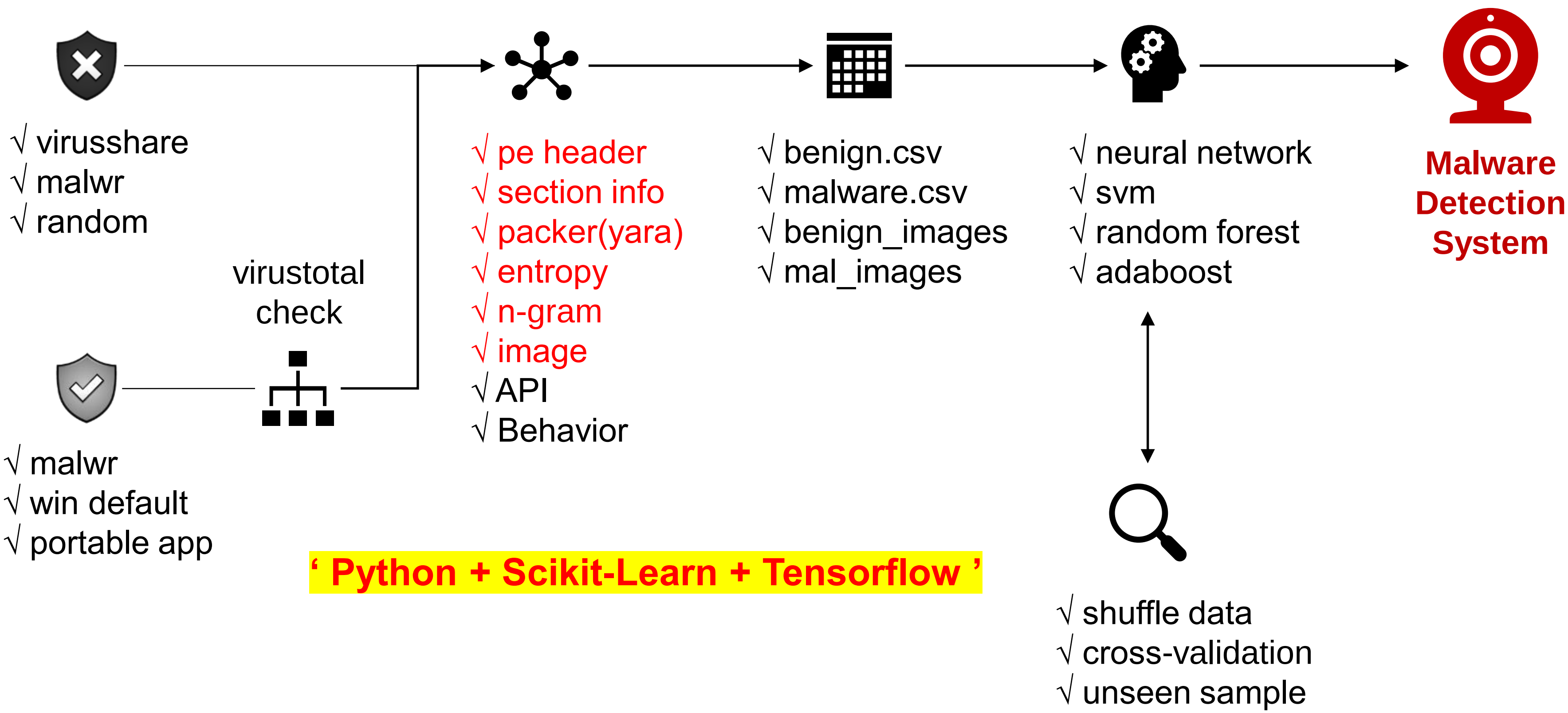
Can you find a sniper ?!

Adversarial Environment

- build own features, parameters, models as many as possible
- As if adversary has knowledge of '4 key factors' (white-box)
- Only validation process is done in black-box environment



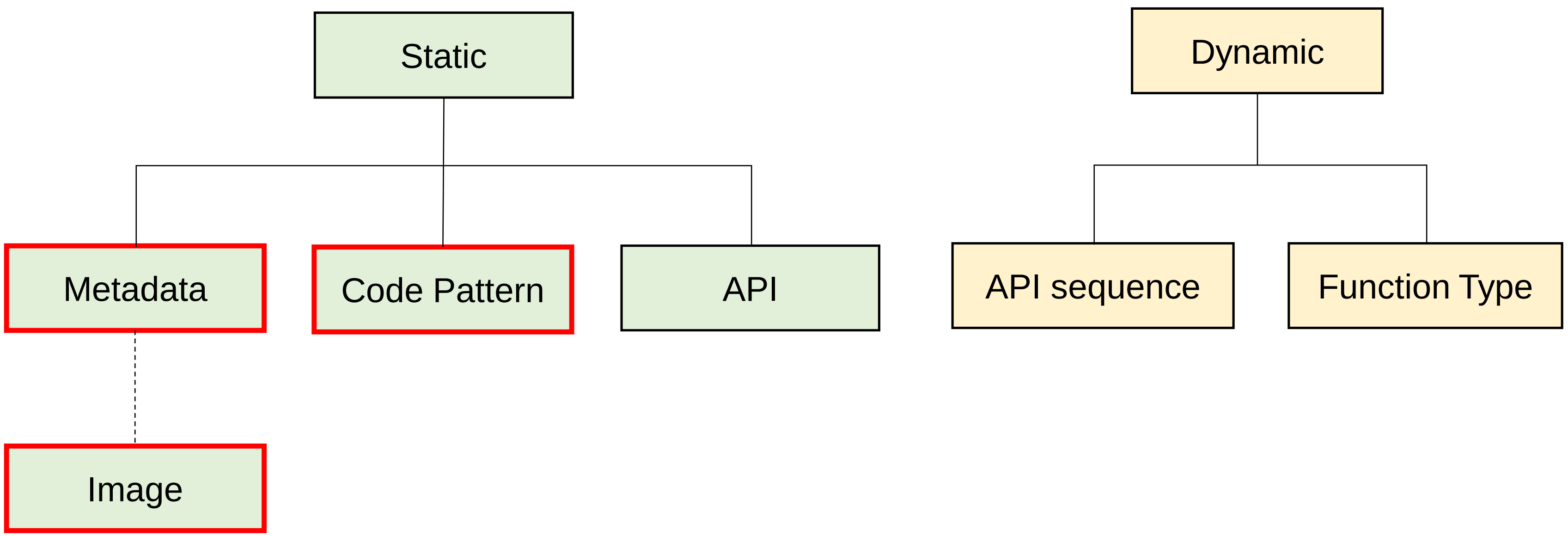
Malware Detection System



It will be uploaded to Github soon !

Feture Extraction

- Only focused on 32-bit PE malwares



Future Extraction

- **Metadata**
  - PE header + Section information
  - Total 68 features → Thanks to ClaMP(https://github.com/urwithajit9/ClaMP)
  - originally 69 features, 69th is 'packertype' (one-hot encoding → 173 features)

ClaMP (Classification of Malware with PE headers)

A Malware classifier dataset built with header fields' values of Portable Executable files

Future Extraction

- **Code Pattern**
  - extract code pattern from disassembled code ← ‘code’ section
  - using n-gram analysis used in text-mining area: 4-gram

1-gram

```
1: {mov,  6}
2: {int,  3}
3: {loop, 1}
```

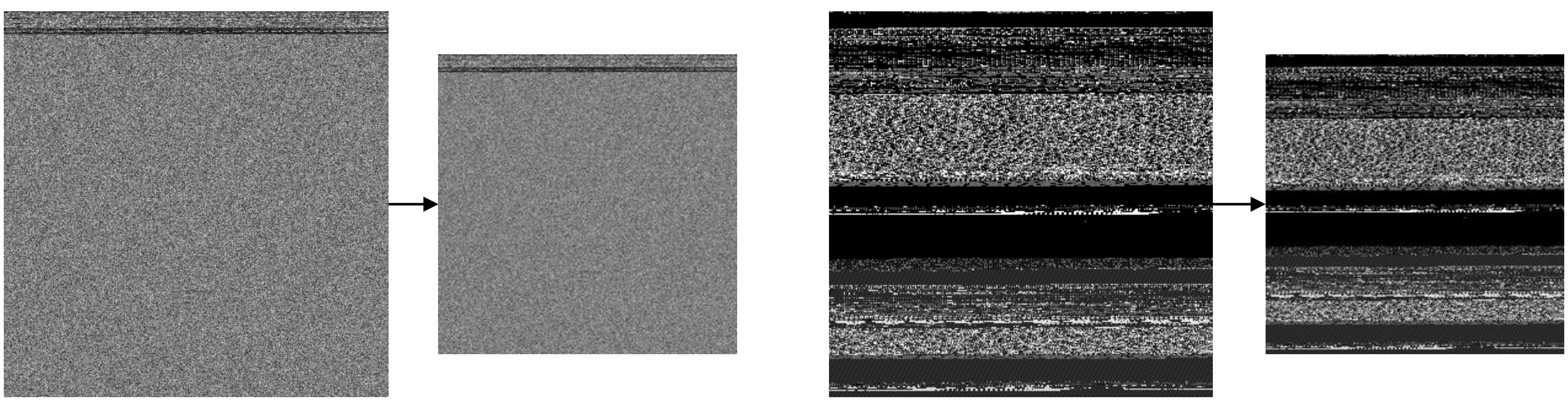
```
mov cx ,count
mov dx,13
mov ah,2
int 21h
mov dl,10
mov ah,2
int 21h
loop first
mov ax,4c00h
int 21h
```

4-gram

```
1: {mov mov mov int, 1}
2: {mov mov int mov, 1}
3: {mov int mov mov, 1}
3: {int mov mov int, 1}
3: {mov mov int loop,1}
3: {mov int loop mov,1}
3: {int loop mov int,1}
```

Future Extraction

- **Image**
  - PE file into image (gray scale)
  - file size is different – different image size → make thumbnail : 256 x 256



Modeling

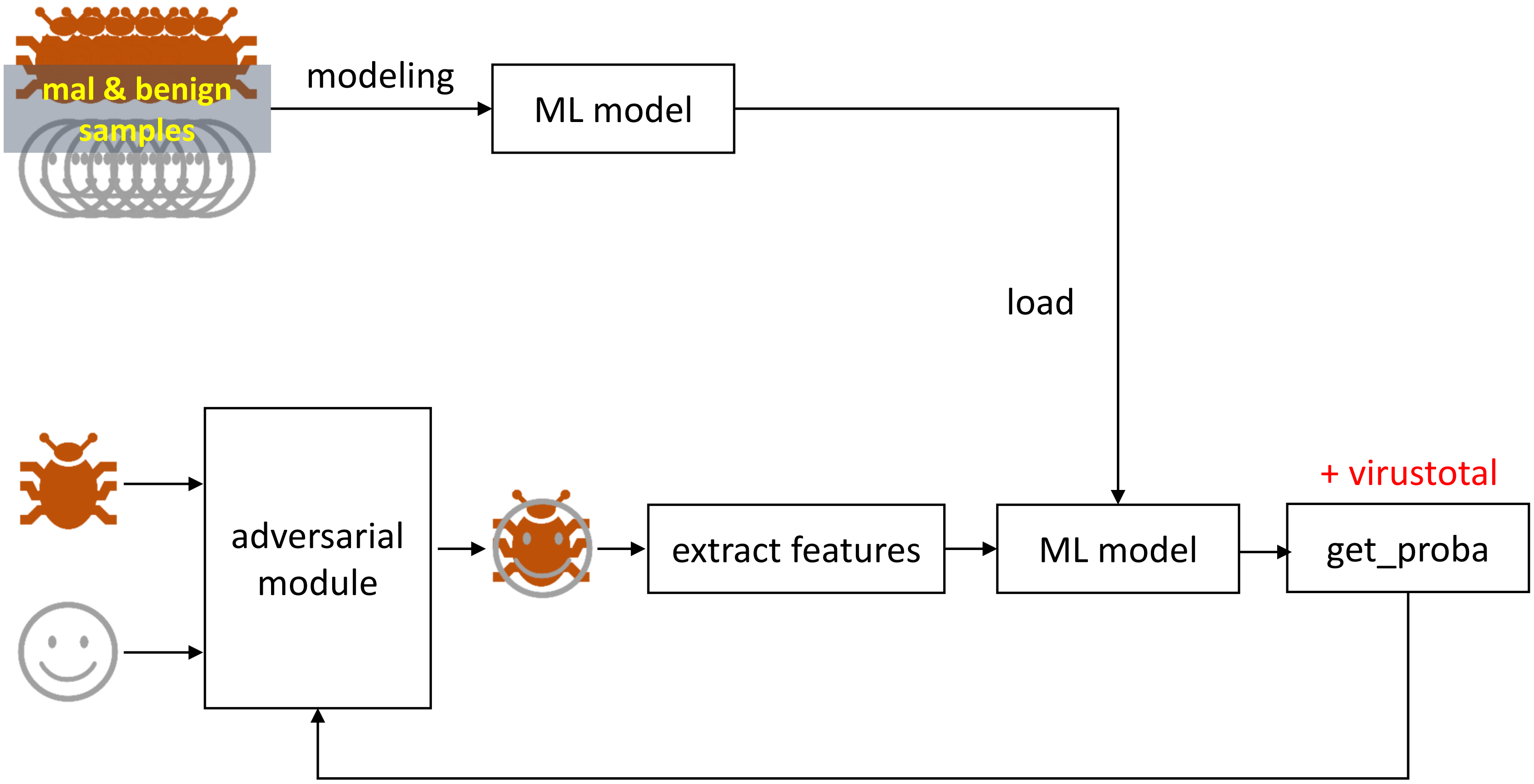
- **Result**
  - Using 10-Fold cross validation
  - 30000 malware samples / 4000 benign samples
  - Accuracy

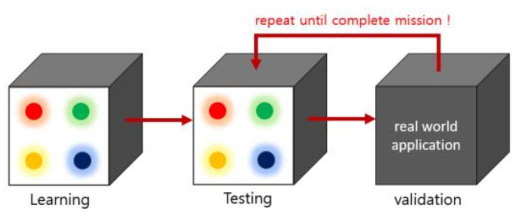
(80 / 20)	n.feat	SVM	R.F	Ada	DNN	CNN
PE	68	91.3 %	97.5 %	95.7 %	92.8 %	-
PE + Packer	173	91.8 %	99.8 %	99.8 %	93.8 %	-
N-gram	10000	87.3 %	99.9 %	100 %	100 %	-
Image	28 x 28	-	-	-	-	99.8 %

1024 deep x 4 layer

MY TARGET !

Attack Scenario





Attack Process

- **Target : randomforest and CNN(deep learning) model**

1. Get probability of sample RandomForest

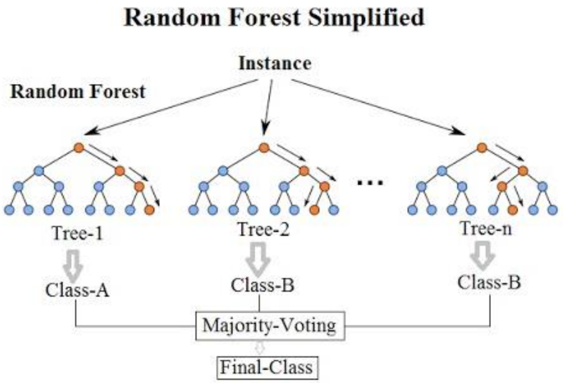
2. Get feature importance from randomforest

3. Feature analysis (divided into 4 class)

4. Overwrite output features and find critical point

5. Disguise a malware as a benign sample

6. Validation



Predict_proba

- scikit-learn provides **predict_prob** ← predict class probabilities
- adversary can estimate the impact of modification using this function

```python
def check_av(test, model):

    if model == 'rf':
        clf = joblib.load('./rf_model.pkl')
        prob = clf.predict_proba(test)
```

```
(adv) C:₩Users₩nabab₩Desktop₩Secuinside₩dataset>python 5_mini_av.py C:₩Users₩nabab₩De
aset₩procexp.exe C:₩Users₩nabab₩Desktop₩Secuinside₩dataset₩mal_new.exe
144,3,4,65535,184,280,5,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,12,0,763904,1929728,0,637496,
1,0,0,5,1,1,1,2727685,2,1,0,1,0,0,0,0,1,0,0,0,1048576,4096,1048576,4096,1,1,4,0,6.47
15,2710688,6.22477709774,1
144,3,4,65535,184,248,3,0,1,1,1,1,0,0,0,1,0,0,0,0,0,0,0,12,0,763904,1929728,0,261728,
1,0,0,5,1,1,1,2727685,2,0,0,0,0,0,0,0,0,0,0,0,1048576,4096,1048576,4096,1,0,3,1,7.51
28,178736,4.45898294652,1
[[ 25.  75.]]
It is MAlware! bro
```

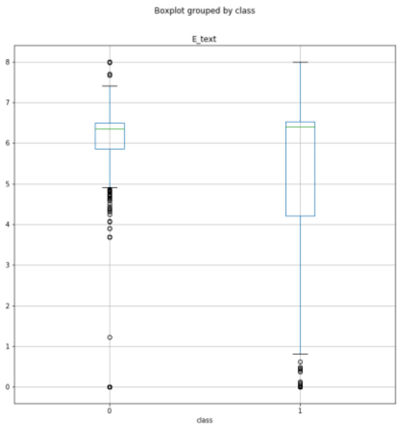
Feature Importance

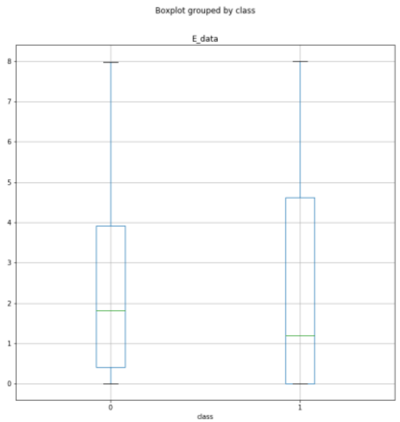
- using randomforest, you can get feature importance of all features
- there is no principle feature → top1 feature only has 12% importance
- so, just top 20 features are used for disguise

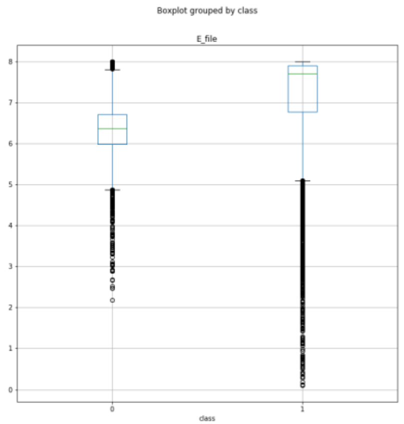
```
1. feature 43 (0.125337). colname Subsystem
2. feature 66 (0.087313). colname E_file
3. feature 42 (0.065199). colname CheckSum
4. feature 7 (0.051913). colname CreationYear
5. feature 65 (0.048669). colname filesize
6. feature 28 (0.044226). colname AddressOfEntryPoint
7. feature 63 (0.039342). colname E_text
8. feature 64 (0.039096). colname E_data
9. feature 25 (0.038991). colname SizeOfCode
10. feature 30 (0.037704). colname BaseOfData
11. feature 38 (0.036354). colname MajorSubsystemVersion
12. feature 26 (0.035825). colname SizeOfInitializedData
13. feature 23 (0.030229). colname MajorLinkerVersion
14. feature 27 (0.029814). colname SizeOfUninitializedData
15. feature 67 (0.029394). colname fileinfo
16. feature 34 (0.025550). colname MajorOperatingSystemVersion
17. feature 5 (0.025085). colname e_lfanew
18. feature 55 (0.019832). colname SizeOfStackReserve
19. feature 6 (0.019508). colname NumberOfSections
20. feature 61 (0.017546). colname non_sus_sections
```

Feature Analysis

- draw histogram, boxplot from all feature vectors
- categorize features into four classes and compare them witth importance data
  - distribution almost same / different number of outlier → 9 / 18
  - different distribution → 4 / 7
  - similar distribution → 7 / 19
  - ~~almost same → 0 / 24~~





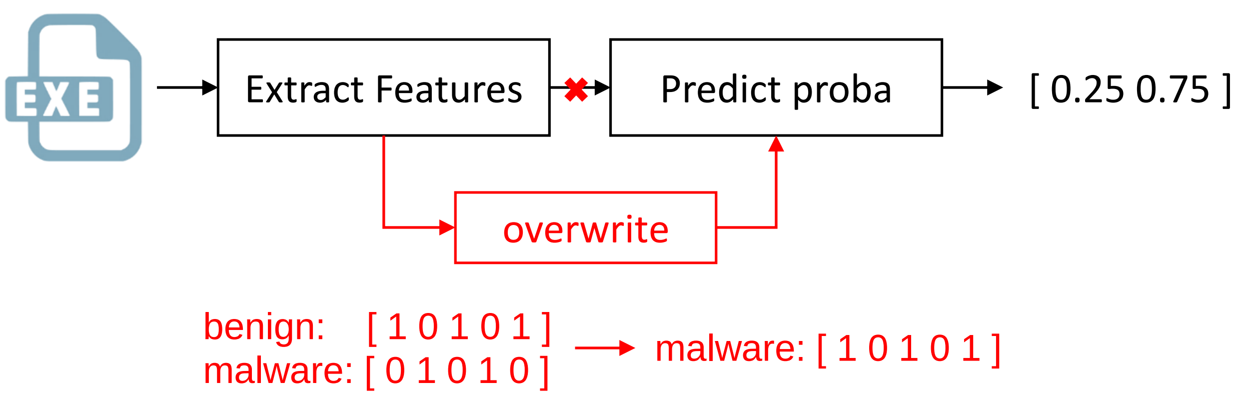


Overwrite Headers

- Just overwrite feature array(benign → malware) by each class from feature analysis

benign malware

- for 100 percent probability malware sample (0 : 100)
  - just one class – probability changed to 90 % (10 : 90)
  - two class – probability does not changed (still 10: 90)
  - three class – probability dropped to 35% (65 : 35) **← bypass classifier !**



File modification

- overwrite extracted features ← meaningless!

- need to change the binary itself
  - **ok to overwrite** (39) – timestamp, checksum, characteristics, linker info, etc
  - **need to care specifications** (5) - entropy of whole file, sections, entrypoint, filesize
- After overwrite features from benign sample into malware sample (39 features)
  - Probability dropped 15 % (0 : 100 → 15 : 85)
  - VirusTotal result : 38 → 32 (what the ?!)

File modification

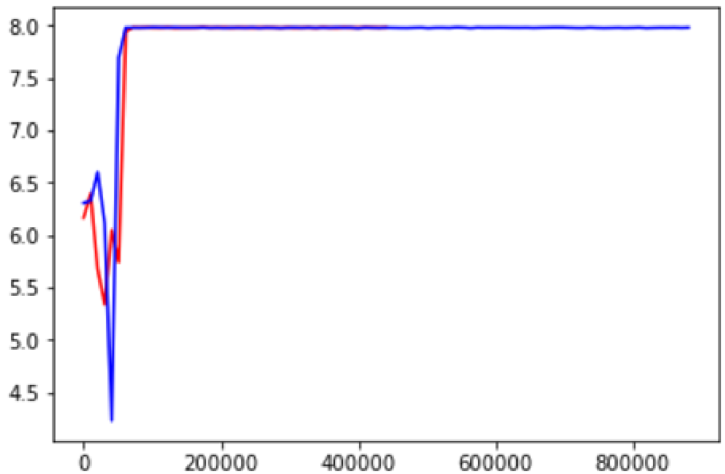
- I just wrote adversarial attack code for my own ML model, but ?!
- decided to keep checking the virustotal report ☺

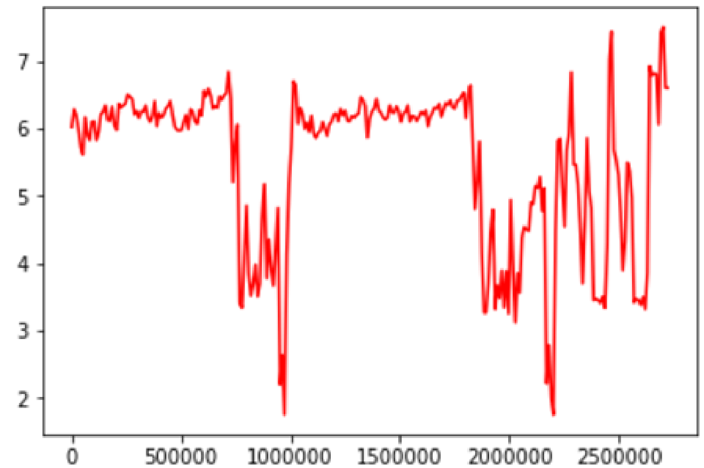
SHA256:	8c8ea74945639026be60ccab5ba463a165b40516ce70d2a55e8e70fd7fd44880
파일 이름:	malware.exe
탐지 비율:	37 / 60
분석 날짜:	2017-06-27 13:22:17 UTC (0분 전)

SHA256:	52d89661bb6403b946a4d5f146a9be2300a1ece93430ff569d8b6f7b82ced693
파일 이름:	mal1.exe
탐지 비율:	32 / 61
분석 날짜:	2017-06-27 13:21:13 UTC (1분 전)

File modification

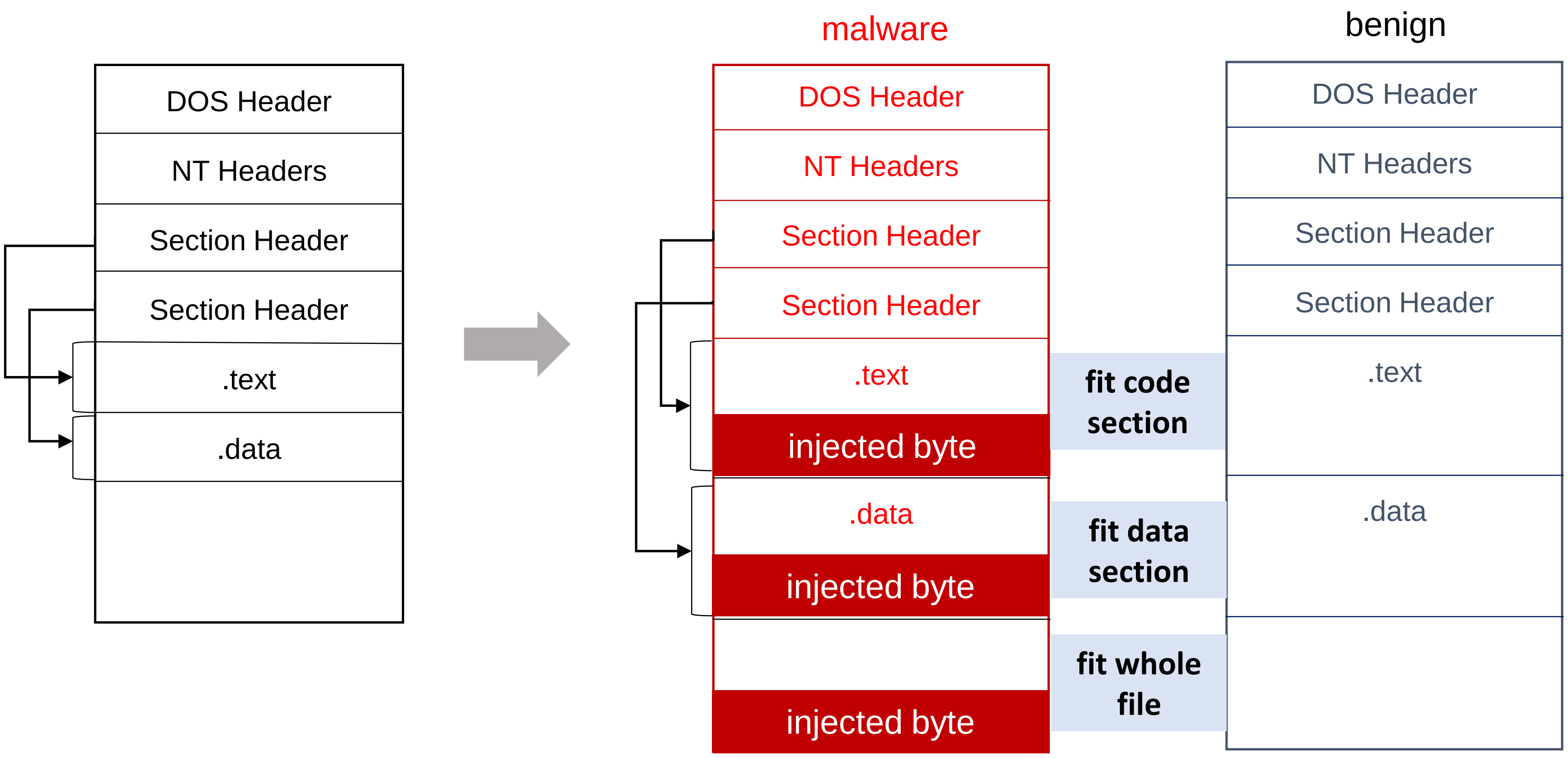
- **Entropy** is a measure of unpredictability of the state, or equivalently, of its average information content
- Entropy of file or specific section can be used as a feature for ML
- It's not a simple job to change entropy of a binary





File modification

- fit malware’s entropy to benign sample



File modification

- After changed both 39 features info + entropy
- Virustotal detection dropped to 26 !

SHA256:	8c8ea74945639026be60ccab5ba463a165b40516ce70d2a55e8e70fd7fd44880
파일 이름:	malware.exe
탐지 비율:	38 / 61
분석 날짜:	2017-07-02 15:37:17 UTC (0분 전)

SHA256:	16174333d6a9183a8f78815e1f1d5f3c105834c268256ac6bdb9aefc5c7586ed
파일 이름:	mal_new.exe
탐지 비율:	26 / 61
분석 날짜:	2017-07-02 15:38:12 UTC (0분 전)

File modification

- Actually, I didn't count the impact of API malware used
- I'm curious, so packing the malware and same test again

SHA256:	175a67a36835740416f3813ac1ff1289ebf09c82ce4ab3e920d7be044a31acfe
파일 이름:	malware_upx.exe
탐지 비율:	$\boxed{25 / 61}$ ← **detection rate dropped after simply packed original file**
분석 날짜:	2017-07-02 15:46:30 UTC (0분 전)

SHA256:	548658009b8ca21a8bd98c40271337208988664bed92610a87f022963cef3dec
파일 이름:	mal_new.exe
탐지 비율:	$\boxed{22 / 61}$ ← **adversarial attack on packed file**
분석 날짜:	2017-07-02 15:47:34 UTC (0분 전)

Model Validation

- then, what about ‘wannacry’ malware sample ?!
- pick a random sample from my dataset and query to virustotal

SHA256:	3dcbb0c3ede91f8f2e9efb0680fe0d479ff9b9cd94906a86dec415f760c163e1
파일 이름:	wanna
탐지 비율:	56 / 60
분석 날짜:	2017-07-03 00:11:27 UTC (1분 전)

- ok, let’s start ☺

Model Validation

- **first step >** after pass the binary to adversarial model (benign: procexp.exe)

SHA256:	2a77be5e16763bedf46919edf22267f533465340aa9f213c28aaa2f1279698c3
파일 이름:	mal_new.exe
탐지 비율:	39 / 61
분석 날짜:	2017-07-03 00:13:03 UTC (0분 전)

- **second step >** pass the binary(from first step) to my ML model

```
(adv) C:₩Users₩nabab₩Desktop₩Secuinside₩dataset>python 5_mini_av.p
aset₩procexp.exe C:₩Users₩nabab₩Desktop₩Secuinside₩dataset₩mal_new
144,3,4,65535,184,280,5,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,12,0,7639
1,0,0,5,1,1,1,2727685,2,1,0,1,0,0,0,0,1,0,0,0,1048576,4096,104857
15,2710688,6.22477709774,1
144,3,4,65535,184,248,4,0,1,1,1,1,0,0,0,1,0,0,0,0,0,0,0,12,0,7639
,0,0,5,1,1,1,2727685,2,0,0,0,0,0,0,0,0,0,0,1048576,4096,1048576
,278576,6.30483512142,1
63  mal:  6.51467471024  ben:  6.47702255325
64  mal:  2.7421050432  ben:  2.69213215915
66  mal:  6.30483512142  ben:  6.22477709774
[[ 20.   80.]]
It is MAlware! bro
```

couldn't bypass my ML model ☹

Model Validation

- **third step >** upx packing and adversarial model

```
aset₩procexp.exe C:₩Users₩nabab₩Desktop₩Secuinside₩dataset₩mal_new.exe
144,3,4,65535,184,280,5,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,12,0,763904,192
1,0,0,5,1,1,1,2727685,2,1,0,1,0,0,0,0,1,0,0,0,1048576,4096,1048576,4096
15,2710688,6.22477709774,1
144,3,4,65535,184,248,3,1,1,1,1,1,0,0,0,1,0,0,0,0,0,0,0,11,0,194560,403
,0,0,5,1,1,1,620899,2,0,0,0,0,0,0,0,0,0,0,0,1048576,4096,1048576,4096,1
,178736,4.45900084397,1
63  mal:  7.55145641137  ben:  6.47702255325
64  mal:  2.66181969728  ben:  2.69213215915
66  mal:  4.45900084397  ben:  6.22477709774
[[ 25.  75.]]
It is MAlware! bro
```

still... malware

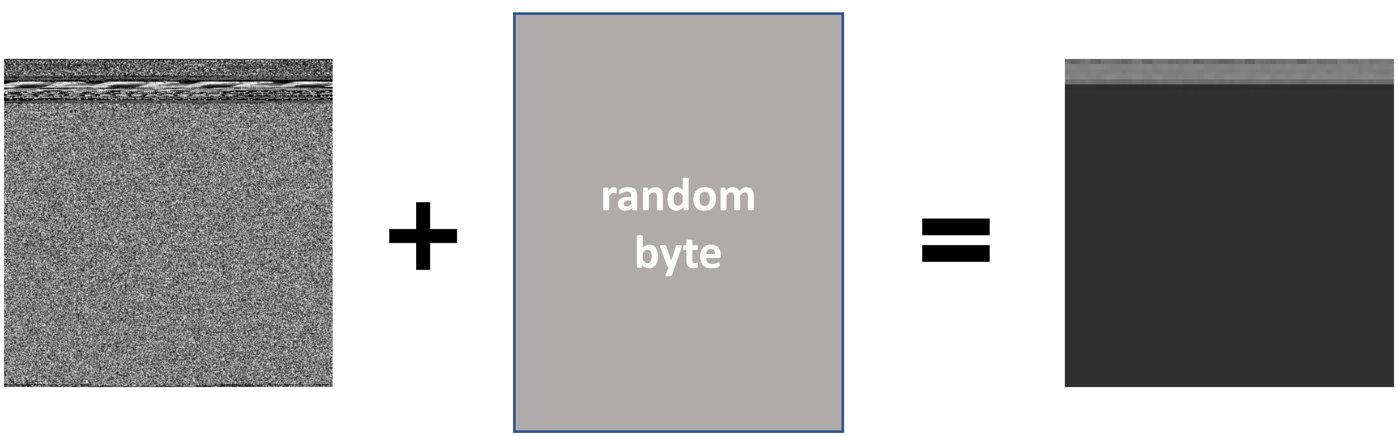
- **fourth step >** query to the virustotal (upx + adv)

SHA256:	4fa41890ace494568f5c614ae05976a0690e2fbcf5c659c19aa072ec1fe76532
파일 이름:	mal_new.exe
탐지 비율:	14 / 61 ← 탐지 비율: 56 / 60
분석 날짜:	2017-07-03 00:19:43 UTC (0분 전)

???? BYPASSED ML BASED AV ~ yeah~ ☺

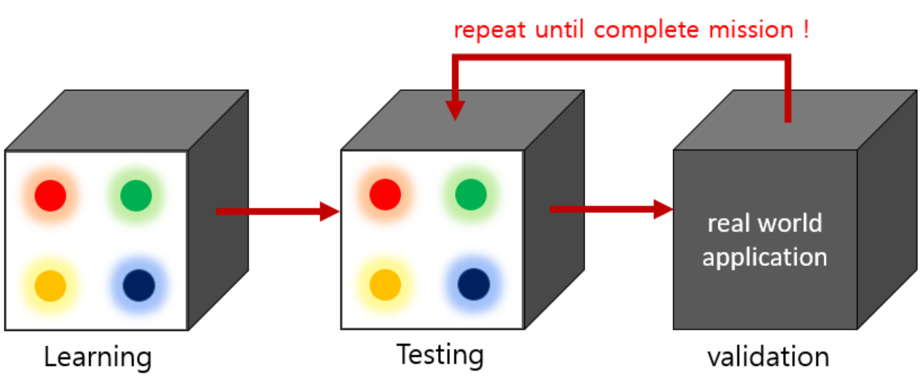
Adversarial Deep Learning

- **If AV use deep learning to classify malware**
- **Candidate model**
  - **DNN –** nothing different than other machine learning algorithm (just deep neural network)
  - **CNN – using binary image as features**
  - **RNN(api sequence)** – using behavior analysis, extract api sequence info from executing
- **main idea** – add garbage byte to the end of the binary. That’s it !



Summary

- develop adversarial model just using static features (PE metadata)
- even build your own model → doesn't tell you the exact answer
- UPX can be used as a camouflage tool
- extract as many features as you can → lead to robust adversarial model
- adversarial model can affect traditional av software (signature based)



Future work

- **Expand Feature Vector** – API, behavior information
- **Reinforcement Machine Learning Model** – automatic adversarial attack
- Virustotal Detection Rate '**Zero**'
- **Develop adversarial testing framework for anti-virus software**

References

- Can Machine Learning Be Secure? - Marco Barreno et al
- Adversarial Machine Learning – J.D. Tygar
- Adversarial and Secure Machine Learning – Huang Xiao
- Adversarial Reinforcement Learning – William Uther et al
- Adversarial Machine Learning – Ling Huang
- Adversarial Examples in the physical world – Alexey Kurakin et al
- Adversarial Examples in Machine Learning – Nicolas Papernot
- Explaining and harnessing adversarial examples – Ian J. Goodfellow et al
- Machine Learning in adversarial environments – Pavel Laskov et al

Thank you

any question? nababora@naver.com