

# How to build a malware classifier

[that doesn't suck on real-world data]

John Seymour

[seymour1@umbc.edu](mailto:seymour1@umbc.edu), @\_delta\_zero

2016-10-18



whoami

- Ph.D. student at the University of Maryland, Baltimore County (UMBC)
- Senior Data Scientist at ZeroFOX, Inc.



# Outline

- Introduction to Malware Classification
- Overfitting and why it's an issue
- Preliminary work- showing classifiers overfit
- How do we fix this?



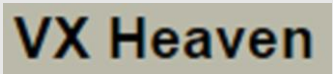



# Malware meets Machine Learning

Main Problem: more malware variants created than we can possibly ever analyze

# Crash Course in Machine Learning

- Machine Learning: finding patterns in data
- Features: these potential patterns
- Models: methods for making sense of features
- Libraries exist for creating models from features (python, R, WEKA)
- Places to get started:
  - <http://www.r2d3.us/visual-intro-to-machine-learning-part-1/>
  - <https://www.dataquest.io/mission/74/getting-started-with-kaggle/>

# Where to find malware

	600 samples	<ul style="list-style-type: none"><li>• Lots of exploit kits</li><li>• Includes analyses</li></ul>
	10,868 samples (about 500GB)	<ul style="list-style-type: none"><li>• 9 families of malware</li><li>• Hexdumps/Assembly files (from IDA)</li><li>• Neutered: PE headers removed</li></ul>
	271,092 samples	<ul style="list-style-type: none"><li>• Labeled by KAV</li><li>• Last update: 2007</li><li>• Most-used academic dataset</li></ul>
	24,783,626 samples	<ul style="list-style-type: none"><li>• Split into chunks of 65,536 samples</li><li>• Available by Torrent</li><li>• Recently labeled (by us!)</li></ul>
 	As many as you want	<ul style="list-style-type: none"><li>• VirusTotal: Needs Private API<ul style="list-style-type: none"><li>• Research Requests</li><li>• Licensing issues</li></ul></li></ul>

# Features commonly used

## PE-File metadata

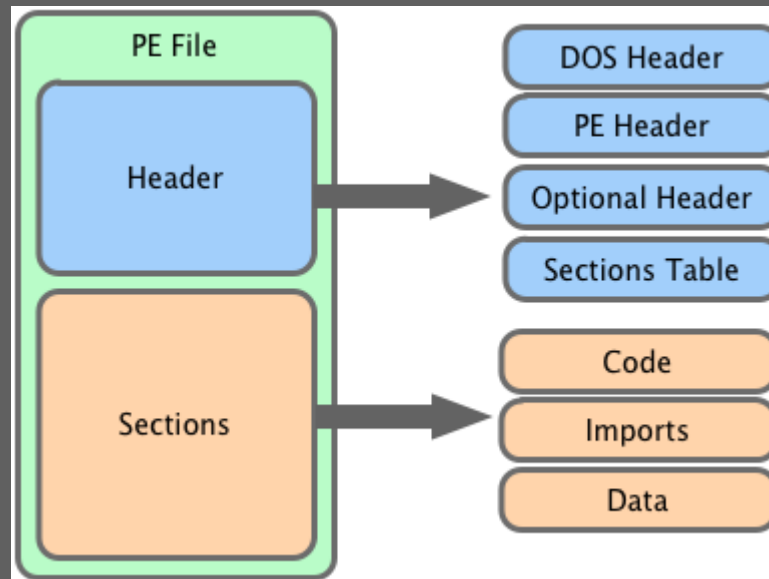


Image courtesy of [trustwave.com](https://trustwave.com)

Python pefile library is amazing

# N-Grams on hexdumps/assembly files

- Sliding window over text

*DEADBEEF*

*DEADBEEF*

*DEADBEEF*

- Features:
  - DEAD: 1
  - ADBE: 1
  - BEEF: 1



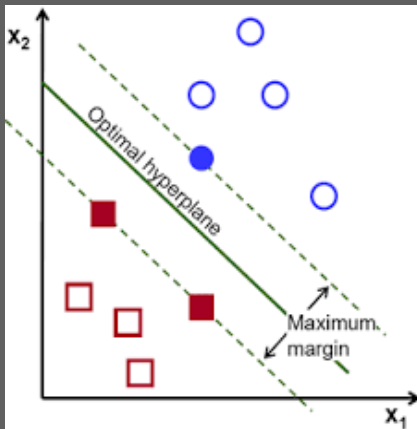
# Other features commonly used

- Opcodes, imports, etc
- Assembly instructions

```
mov     eax, ebp
call    ClassAlloc13680_0FAh ; (eax=this,edx=this,ecx=83, ebx=118, arg1=193, arg2=60, arg3=0x0, arg4=2, arg5=0)
push    0
mov     edx, [eax]
push    2
mov     ecx, 21h
mov     [edx+Class180F4.WidgetInputHandler], offset gblHandleTransportDestinationAndCheckForPassengersSpace
push    0
mov     edx, [eax]
mov     ebx, 5Ch
push    4Eh
mov     [edx+Class13680.Paint???], offset ClassVehicleManager__PaintForSomeWidget
mov     dword_4088, eax
mov     eax, [eax]
push    5Bh
mov     edx, ebp
mov     [eax+Class10B8C.MouseInputHandler], offset ClassVehicleManager__MouseInputHandler
mov     eax, ebp
call    ClassAlloc13680_0FAh
push    0
push    2
```

# Top performing models

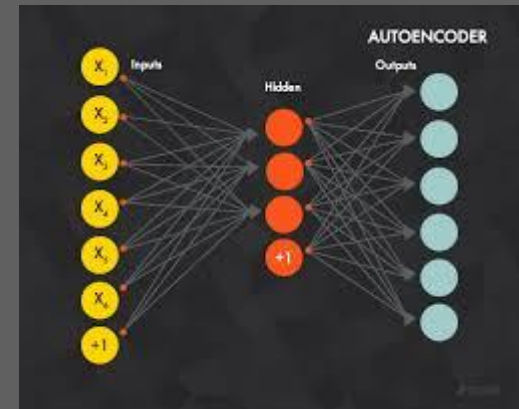
SVMs



xgboost



Deep Learning



Again, use libraries!

## Early Works

- Kolter and Maloof (2006)
  - N-Grams of byte code
  - ~3600 total malware/benign instances
- PE Miner (2009)
  - PE header information
  - ~17,000 total malware/benign instances
  - Defined ML success criteria
  - However, dataset/final features are not public
    - Limited reproductive power

## Adobe Malware Classifier (2012)

- Presented at BlackHat USA/InfoSec SouthWest
- Claimed a 98.56% detection rate
- Used only 7 features
- 116,000 total malware/benign specimens
  - Malware was from Vx Heaven
  - Benign was from clean Windows 7 installation
- Completely open source
  - Even had a python tool for classifying new .exe's



Completed • \$16,000 • 377 teams

## Microsoft Malware Classification Challenge (BIG 2015)

Tue 3 Feb 2015 – Fri 17 Apr 2015 (14 months ago)

You are provided with a set of known malware files representing a mix of 9 different families. Each malware file has an Id, a 20 character hash value uniquely identifying the file, and a Class, an integer representing one of 9 family names to which the malware may belong:

1. Ramnit
2. Lollipop
3. Kelihos\_ver3
4. Vundo
5. Simda
6. Tracur
7. Kelihos\_ver1
8. Obfuscator.ACY
9. Gatak

## Overfitting, and why it's an issue

Overfitting is when your machine learning model doesn't generalize to new instances

This is bad, because we want to use our models to classify new stuff

Goal here today: convince you that these previous models overfit, even if they attempted to avoid it

# Adobe Classifier Overfit?

Hint: Only 7 features for 98% accuracy!

Hint: Only Windows executables for benign data

Hint:

1) *DebugSize*. Denotes the size of the debug-directory table. Usually, Microsoft-related executable files have a debug directory. Hence many clean programs may have a non-zero value for *DebugSize*

## Adobe Classifier Overfitting Test (2015)

- Downloaded 542 benign executables from Cygwin/SourceForge
- Tested model on new executables
- Classifier guesses: 462 malicious, 71 UNKNOWN, 6 break the script, 3 benign





## Discussion

- We did go back and check to make sure our SourceForge executables were actually benign
- Adobe experiment used more training data than Kolter and Maloof/PE-Miner...

# Kaggle Overfitting Dataset (BSidesLV 2016)

- Goal: how does the top Kaggle model perform on a new dataset?
- Step 1: Find similar executables somewhere else

1. Ramnit
2. Lollipop
3. Kelihos\_ver3
4. Vundo
5. Simda
6. Tracur
7. Kelihos\_ver1
8. Obfuscator.ACY
9. Gatak

# Why VirusShare?

- Pro: Size (27 million samples: almost 2500x Kaggle dataset size, not neutered)
- Pro: Consistently updated
- Pro: Makes future ML research more reproducible
  - Scrape your own => you'll probably overfit
  - VirusTotal: can't release any raw data from the platform
  - Short descriptions of dataset: "Chunks 25, 60, 90"
- Con: Unlabeled

# We can fix that!

- VirusTotal has an awesome API
- Two versions: Private/Research and Public
- Public is rate-limited
- Private/Research has licensing agreements
- Meaning people can't distribute results

```
# Given a batch of hashes, retrieve latest analyses for those hashes
# to VirusTotal in one HTTP request.
# Straight from VirusTotal API documentation, except for retry decorator
@retry(stop_max_attempt_number=5)
def retrieve_batch(batch, user):
    url = "https://www.virustotal.com/vtapi/v2/file/report"
    with open(user, 'r') as key_file:
        api_key = key_file.readlines()[0].strip()
    resource_str = ','.join(batch)
    parameters = {"resource": resource_str,
                  "apikey": api_key}
    data = urllib.urlencode(parameters)
    req = urllib2.Request(url, data)
    response = urllib2.urlopen(req, timeout = 20)
    results = response.read()
    return results
```

```
{u'md5': u'0295c858052b0338c932728c09e17613',
 u'permalink': u'https://www.virustotal.com/file/bf0c62ca4dcf8d67c6dcd848b33c7eb0d60cb74ac8f9ead1fa50103d4ecab674/analysis/1462677708/',
 u'positives': 29,
 u'resource': u'0295c858052b0338c932728c09e17613',
 u'response_code': 1,
 u'scan_date': u'2016-05-08 03:21:48',
 u'scan_id': u'bf0c62ca4dcf8d67c6dcd848b33c7eb0d60cb74ac8f9ead1fa50103d4ecab674-1462677708',
 u'scans': {u'ALYac': {u'detected': True,
 u'result': u'Gen:Variant.Kazy.36619',
 u'update': u'20160507',
 u'version': u'1.0.1.9'},
 u'AVG': {u'detected': True,
 u'result': u'Win32/Cryptor',
 u'update': u'20160507',
 u'version': u'16.0.0.4565'},
 u'AVware': {u'detected': True,
 u'result': u'Trojan.Win32.Encpk.abp (v)',
 u'update': u'20160508',
 u'version': u'1.5.0.42'},
 u'Ad-Aware': {u'detected': True,
 u'result': u'Gen:Variant.Kazy.36619',
 u'update': u'20160508',
 u'version': u'3.0.2.1015'},
 u'AegisLab': {u'detected': False,
 u'result': None,
 u'update': u'20160508',
 u'version': u'4.2'},
 u'AhnLab-V3': {u'detected': True,
 u'result': u'Worm/Win32.Palevo',
 u'update': u'20160507',
 u'version': u'2016.05.08.00'},
```

## Results from Labeling

- 30 people + 6 months to label entire corpus
  - Mostly due to rate-limiting
  - Mostly undergraduates wanting extra credit
  - Shoutout to MLSec Project

Labels available here:

[https://drive.google.com/file/d/0B\\_IN6RzP69b2TkNrYVdOMnQ4LVE/view](https://drive.google.com/file/d/0B_IN6RzP69b2TkNrYVdOMnQ4LVE/view)

## Words of warning

- Don't use this to compare AVs
  - VT has a big disclaimer on their site
  - Main reason: not all AVs integrate their full software into VT
- Ground truth might be noisy
  - AV labels sometimes different even though specimens are similar
  - AV labels sometimes similar even when specimens are different.

# What is an index and why do I need one?

- What we have:

Chunk #	Labels
Chunk 0	W32.HfsAdware.1166, PUA.Mindsparki1.Gen, PUP.Optional.MindSpark, MyWebSearch.J (v) (not malicious), Riskware.Win32.Mywebsearch.dsymo, ...
Chunk 1	Win32/Bifrose.D!generic, Gen:Heur.VB.Krypt.13, Heur.Win32.VBKrypt.1!O, Trojan.VB.Gen, Gen:Heur.VB.Krypt.13, Trojan.Injector.Win32.32287, ...
Chunk 2	Trojan.JS.Agent.GCI, Trojan.JS.Agent.GCI, JS.ObfusTools.E, Trojan.JS.Agent.GCI, Troj.Downloader.JS.Twetti.t!c, Exploit ( 04c556691 ), ...
Chunk 3	Win32/SillyAutorun.ABS, Trojan.Generic.615860, Trojan/W32.Agent.39936.W, Generic.Win32.7352f412d0!CMCRadar, Generic.dx!7352F412D0B9, ...
Chunk 4	W32.Generic.Jeefo.Trojan, Win32/Jeefo.A, Win32.Jeefo.B, Virus/W32.Hidrag, Virus.Win32.Hidrag!O, W32.Jeefo.A, Win32.Jeefo.B, Virus.Jeefo, ...

- What we want:

	Count in Chunk 0	Count in Chunk 1	Count in Chunk 2	Count in Chunk 3	Count in Chunk 4
W32.HfsAdware.1166	???	???	???	???	???
PUA.Mindsparki1.Gen, PUP.Optional.MindSpark	???	???	???	???	???
MyWebSearch.J (v) (not malicious)	???	???	???	???	???
Riskware.Win32.Mywebsearch.dsymo	???	???	???	???	???

- Why?
  - Much, much easier to search



# Building an index

- The MapReduce framework is awesome for counting things
  - See PySpark tutorials

# Actual output of index

```
Label,000,001,002,003,004,005,006,007,008,009,010,011,012,013,014,015,016,017,018,019,020,021,022,02$
Win32.Application.ClientConnectConduitDL.C,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0$
Riskware/WebWatcher,5,7,0,0,5,24,50,76,48,16,4,5,2,3,4,2,3,4,1,4,16,20,27,36,18,11,7,7,16,34,22,9,17$
TR/Poly.Agent.C,6,4,30,0,3,13,4,0,7,7,3,0,1,1,0,1,2,5,5,2,0,1,0,0,0,1,2,0,2,0,1,2,0,0,0,0,0,0,0,1,0,2,$
Trojan.Fraudster.1052,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,$
Trojan/Generic.bedbi,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0$
TrojWare.Win32.Buzus.fvjf,46,17,0,0,1,3,1,0,10,7,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0$
Java/SmsSender,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0$
Autorun.FLT,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0$
W32/InjectorPak.GAZ!tr,7,78,0,1,51,189,58,105,765,502,18,0,3,1,2,1,2,3,3,5,3,3,1,0,1,5,1,0,2,0,1,3,2$
Virus.Win32.Virut.1!0,286,383,1,0,222,972,438,21,20,235,46,0,1,1,1,2,0,13,2,1,17,23,39,47,29,12,14,1$
Troj/BHO-PC,0,0,66,0,0,0,0,1,2,7,0,0,0,0,0,0,0,0,4,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,43,52,56,38,49,53,62,$
Troj/BHO-PX,44,23,0,0,0,1,2,1,5,12,1,2,0,1,2,1,6,36,12,0,0,1,1,0,1,1,1,0,0,0,0,0,0,0,30,26,26,21,19,28$
TrojWare.Win32.Rouge.KDVS,13,2,0,0,3,16,5,7,17,9,14,12,12,14,14,18,10,8,18,13,36,50,56,171,94,85,42,$
WIN.Virus.Expiro-5,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0$
WIN.Virus.Expiro-6,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0$
WIN.Virus.Expiro-9,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0$
Worm.Generic.363411,0,268,0,0,224,663,12,35,410,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,$
HackTool.CheatEngine!/Rvf5d7ZQPY,15,15,7,0,3,8,8,1,9,12,10,13,25,18,16,24,25,19,21,21,21,19,19,22,19$
Koutodoor.gen.l,8,5,0,0,23,52,171,298,9,5,73,174,149,164,175,142,143,124,127,137,110,44,36,27,116,13$
Koutodoor.gen.n,6,33,0,0,163,603,364,8,7,71,33,8,8,7,12,4,6,7,7,10,133,152,83,91,58,42,106,93,261,46$
Koutodoor.gen.b,46,94,0,0,31,218,331,206,66,42,126,249,263,257,236,232,303,243,250,260,199,60,47,79,$
Koutodoor.gen.d,5,22,0,0,2,0,26,30,19,1,7,146,133,163,108,125,89,89,67,136,73,4,26,22,94,88,44,52,49$
Koutodoor.gen.g,150,343,0,0,85,285,220,72,58,69,78,272,234,231,245,251,236,177,198,230,690,829,1000,$
Win32/Adware.Coolezweb,0,2,3,26,19,0,14,25,8,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0$
Trojan.Agent/Gen-Injector[Fmt],0,20,25,0,9,13,98,259,170,54,65,26,23,36,24,17,32,30,26,23,27,33,34,4$
W32.Clod691.Trojan.5867,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,$
Trojan.Generic.KDV.271750,0,0,0,0,5,0,32,52,0,6,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0$
VBTrój.AOSA,1,10,5,1,9,3,8,12,40,5,0,0,1,1,1,0,0,1,0,1,0,1,0,1,1,0,1,1,0,2,1,1,0,16,13,14,19,18,13,1$
Troj/Fosniw-F,159,121,0,0,34,111,171,342,586,273,97,164,177,173,216,195,189,159,181,181,185,131,146,$
PWS-Zbot.gen.hb,12,114,2,0,46,45,29,42,109,62,8,18,6,5,34,18,13,17,23,23,47,70,60,75,33,31,34,10,9,1$
PWS-Zbot.gen.hv,24,57,269,5,289,954,371,45,91,204,44,4,2,2,8,6,5,12,6,7,18,114,126,242,57,67,68,176,$
```

# It's really easy to use!

```
seymour1@REM:~/label/count_labels$ time grep "Vundo" index_removed_lowcounts.csv > test.csv
```

```
real    0m0.057s
user    0m0.027s
sys     0m0.011s
```

[illegible]

## Okay, so now we're ready to test Kaggle Models

1. Extract 13,997 Kaggle family instances  
(Chunks 32, 84, 121, 231, 233 have a balance)
2. Transform those instances into Kaggle format
  1. Use Capstone to programmatically disassemble binaries (IDA Pro is meant for single executables)
  2. Use xxd to generate hexdumps
3. Clone top model
4. Replace Kaggle test set with our alternate one, train, and test!

## Kaggle Overfitting Test (TODAY)

- Top model: “say NOOOOOO to overfittttting”
- Model was 95% confident on average in its predictions on our new test set, regardless of correctness
- ...
- ...
- Model performed worse than random chance on new files (1406 correct/13,997 total)

## Discussion: Dataset Quality

- Maybe data transformations are more important than initially thought?
  - For example, we used Capstone instead of IDA Pro to disassemble
  - But, if this was the main issue, the confidence of the model should go down when evaluating new data...
- Maybe models just need more/better data?
  - Would be good to know!

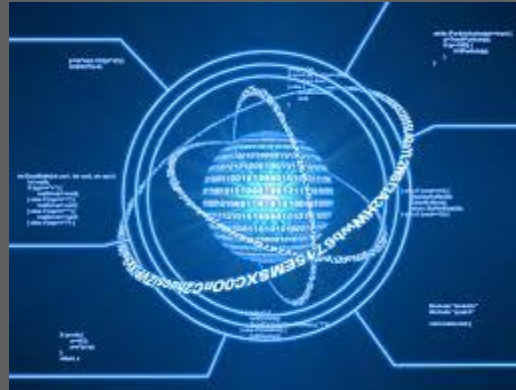
## Discussion: Ground Truth

- Maybe combining AV vendors leads to bad ground truth?
  - After all, we've seen the quality of the labels...
  - 1.5% of VirusShare executables landed in multiple Kaggle families!
  - Microsoft labels don't match up with data they've released!
- There's still no good "benign" dataset

# Key Messages

- Question what “accuracy” means for machine learning models
  - Real-world deployment is best measure of accuracy
- Push for reproducible experiments in scientific discussions





Thank you!

Questions?

[seymour1@umbc.edu](mailto:seymour1@umbc.edu), @\_delta\_zero