Intrusion Detection with Machine Learning

Bryan Solis

Data 606

Recap

- Intrusion Detection System using the KDD-Cup 99 Dataset
- Two NN Models 3 layers and 5 layers
- Adding Object type columns
  - From 43 to 118 columns

What is new ?

Improvements on our NN Model

- Added a monitor
- Multiple test with different layers

Two different models

- Decision Tree
- Random Forest

Accuracy = 0.7709731288902384
Classification Report =

	precision	recall	f1-score	support
back.	0.00	0.00	0.00	441
buffer_overflow.	0.50	0.50	0.50	2
ftp_write.	0.00	0.00	0.00	1
guess_passwd.	0.00	0.00	0.00	9
imap.	1.00	1.00	1.00	6
ipsweep.	1.00	0.07	0.12	260
land.	1.00	1.00	1.00	5
loadmodule.	0.00	0.00	0.00	3
neptune.	0.42	0.00	0.00	21486
nmap.	0.00	0.00	0.00	42
normal.	0.47	0.99	0.63	19630
perl.	0.00	0.00	0.00	1
phf.	0.00	0.00	0.00	1
pod.	0.81	0.98	0.89	57
portsweep.	0.84	0.63	0.72	180
rootkit.	0.00	0.00	0.00	3
satan.	1.00	0.85	0.92	325
smurf.	1.00	1.00	1.00	55919
spy.	0.00	0.00	0.00	0
teardrop.	1.00	0.99	1.00	216
warezclient.	0.95	0.09	0.17	215
warezmaster.	0.50	0.33	0.40	3
accuracy			0.77	98805
macro avg	0.48	0.38	0.38	98805
weighted avg	0.76	0.77	0.70	98805

Decision Tree

Accuracy = 0.9445878245028085
Classification Report =

	precision	recall	f1-score	support
back.	0.00	0.00	0.00	441
buffer_overflow.	0.50	0.50	0.50	2
ftp_write.	0.00	0.00	0.00	1
guess_passwd.	1.00	0.89	0.94	9
imap.	1.00	0.33	0.50	6
ipsweep.	1.00	0.05	0.10	260
land.	0.50	0.80	0.62	5
loadmodule.	0.00	0.00	0.00	3
neptune.	1.00	0.80	0.89	21486
nmap.	1.00	0.31	0.47	42
normal.	0.78	1.00	0.88	19630
perl.	0.00	0.00	0.00	1
phf.	0.00	0.00	0.00	1
pod.	0.92	0.98	0.95	57
portsweep.	1.00	0.63	0.78	180
rootkit.	0.00	0.00	0.00	3
satan.	1.00	0.72	0.84	325
smurf.	1.00	1.00	1.00	55919
teardrop.	1.00	0.06	0.12	216
warezclient.	0.97	0.32	0.48	215
warezmaster.	1.00	0.33	0.50	3
accuracy			0.94	98805
macro avg	0.65	0.42	0.46	98805
weighted avg	0.95	0.94	0.94	98805

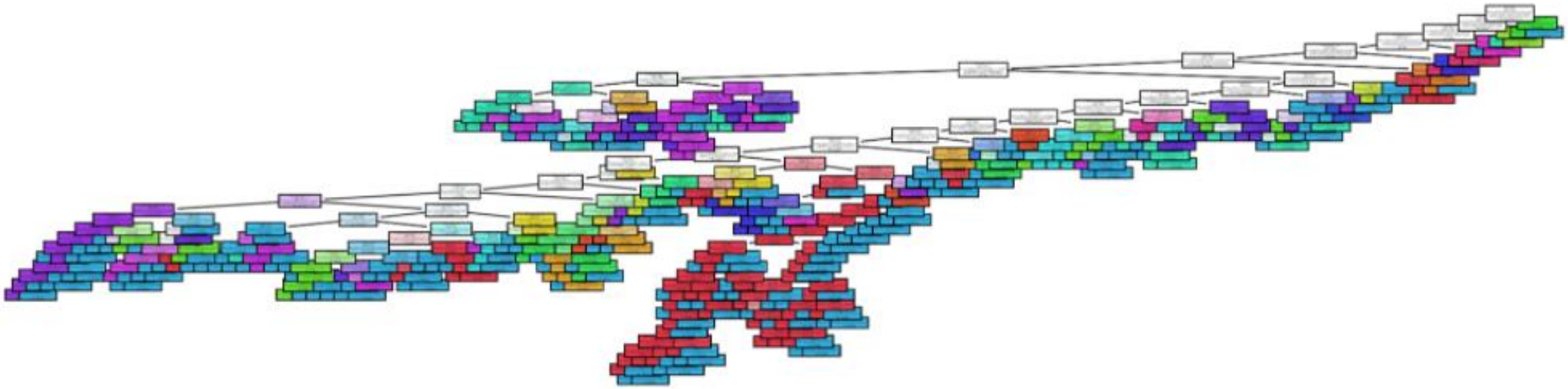
Random Forest

Accuracy = 0.9838469713071201
Classification Report =

	precision	recall	f1-score	support
back.	0.02	0.04	0.03	444
buffer_overflow.	0.00	0.00	0.00	7
ftp_write.	0.00	0.00	0.00	1
guess_passwd.	0.00	0.00	0.00	13
imap.	0.00	0.00	0.00	2
ipsweep.	0.46	0.93	0.62	233
land.	0.00	0.00	0.00	3
loadmodule.	0.00	0.00	0.00	4
multihop.	0.00	0.00	0.00	1
neptune.	1.00	1.00	1.00	21489
nmap.	0.00	0.00	0.00	51
normal.	0.98	1.00	0.99	19498
perl.	0.00	0.00	0.00	1
phf.	0.00	0.00	0.00	2
pod.	0.00	0.00	0.00	55
portsweep.	0.00	0.00	0.00	206
rootkit.	0.00	0.00	0.00	2
satan.	0.00	0.00	0.00	309
smurf.	1.00	1.00	1.00	56093
teardrop.	0.00	0.00	0.00	210
warezclient.	0.42	0.06	0.10	175
warezmaster.	0.00	0.00	0.00	6
accuracy			0.98	98805
macro avg	0.18	0.18	0.17	98805
weighted avg	0.98	0.98	0.98	98805

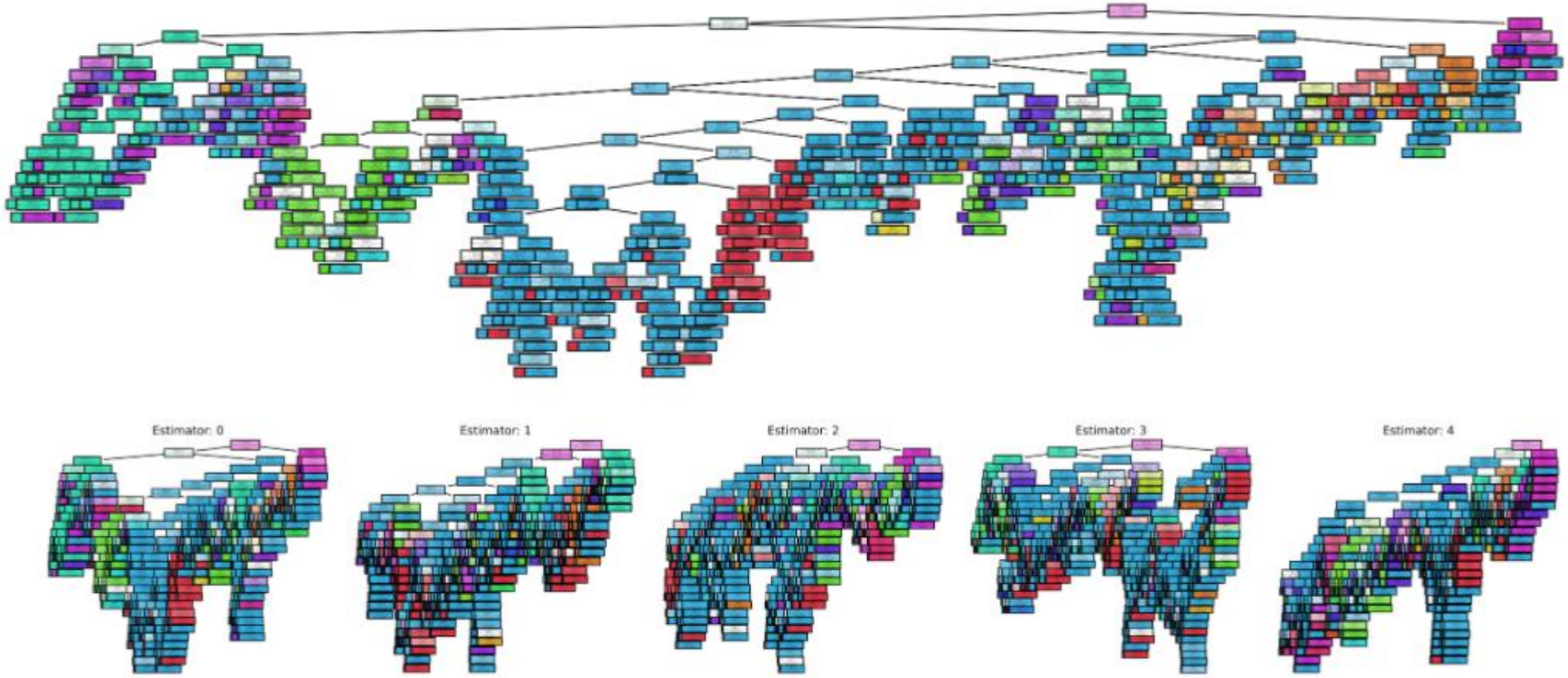
Neural Network

Classification Report for all Models



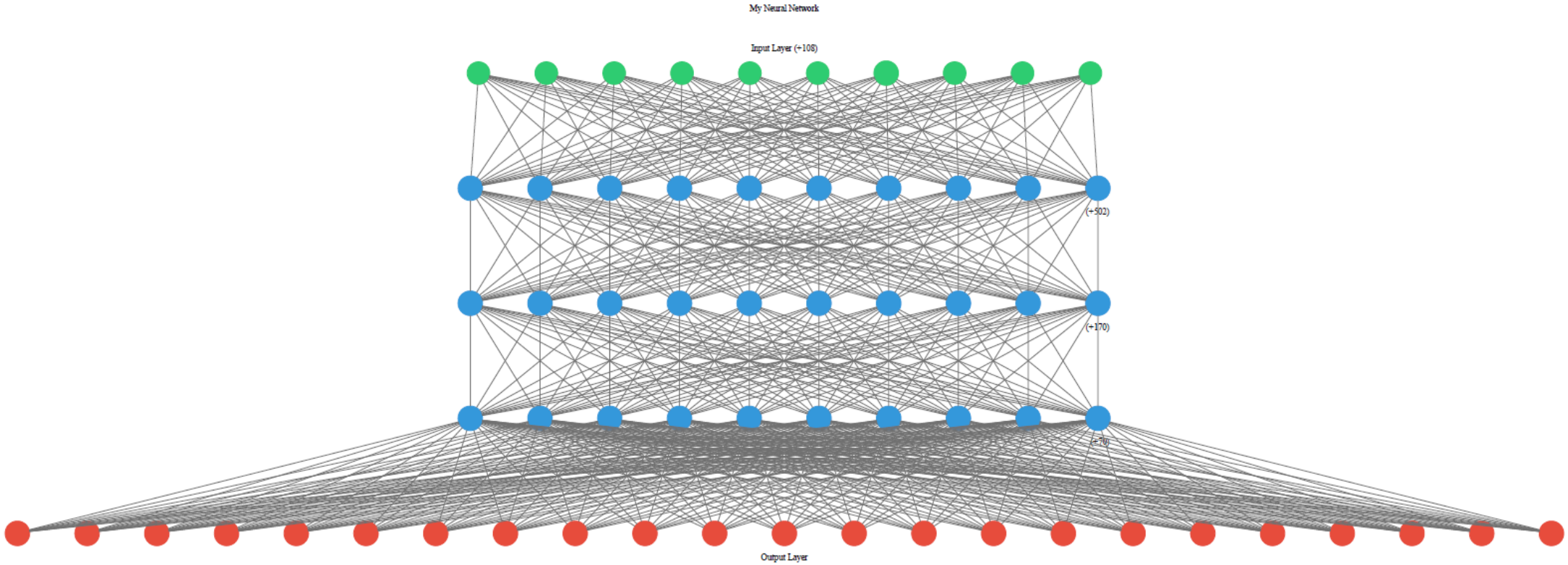
Decision Tree

Accuracy of 77%



Random Forest

Accuracy of 94-95%

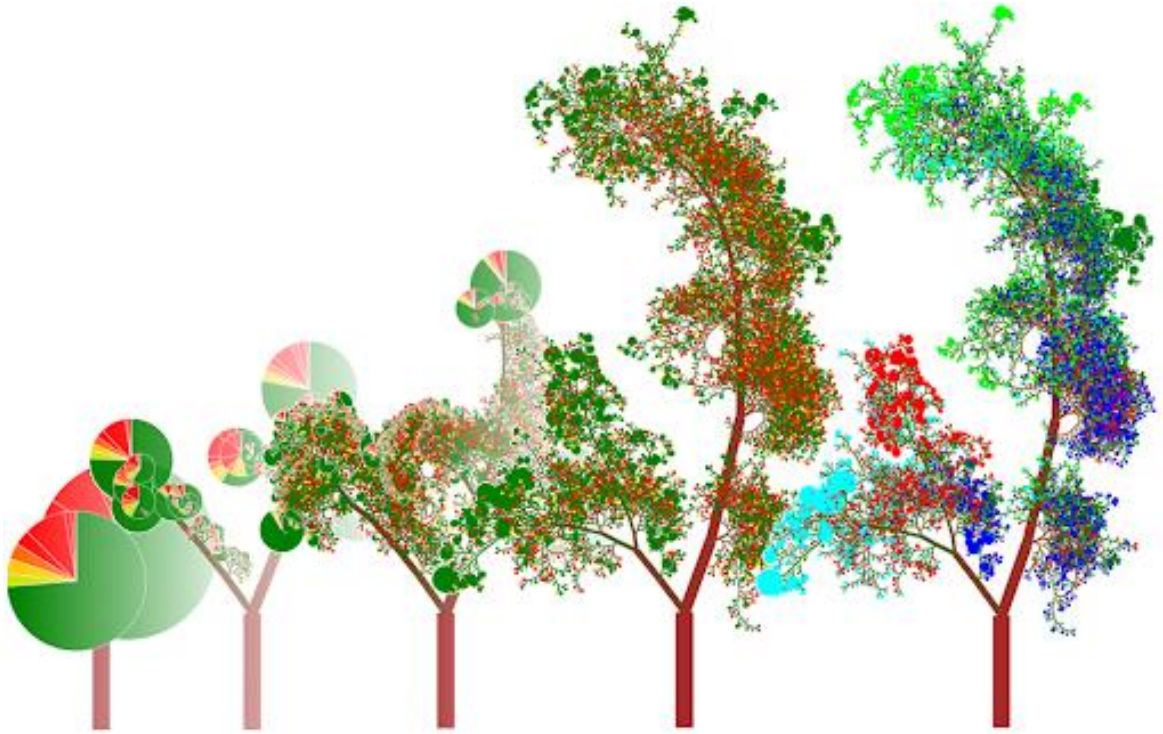


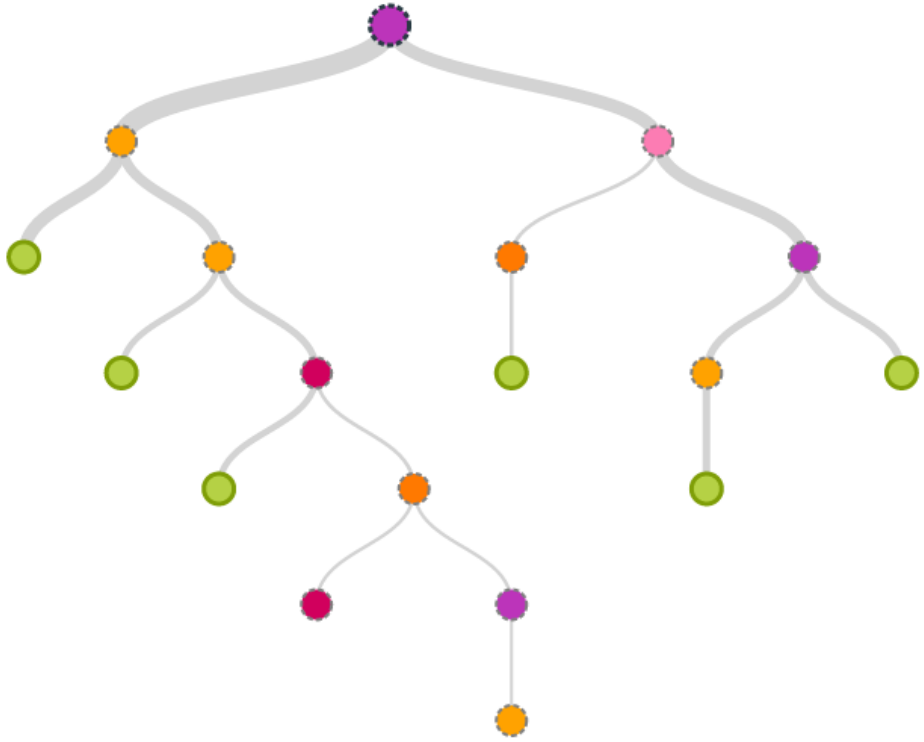
Neural Network

Accuracy 98-99%

Why RF perform better than a Decision Tree ?

- Random forest leverages the power of multiple decision trees.
- It does not rely on the feature importance given by a single decision tree.



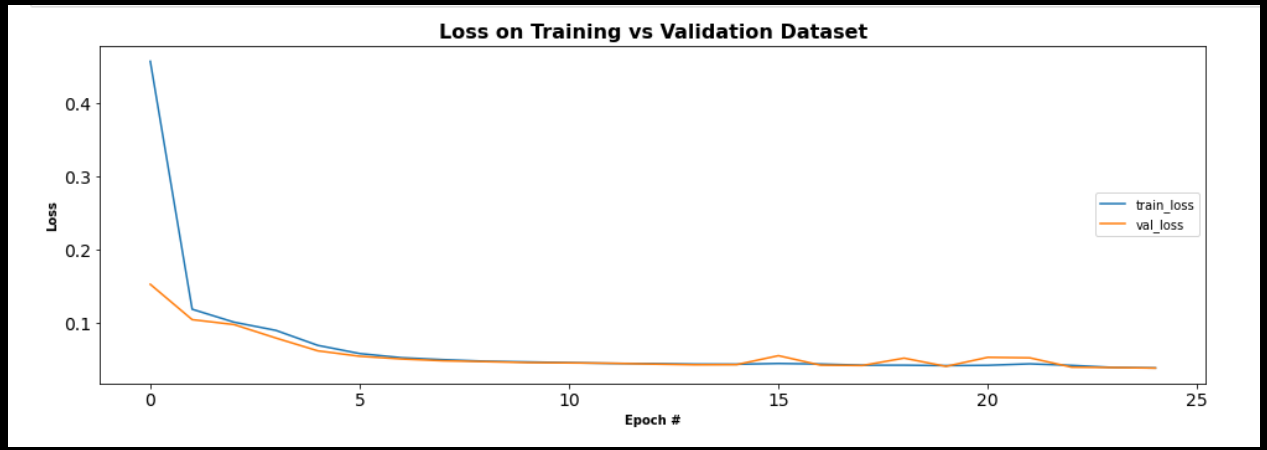


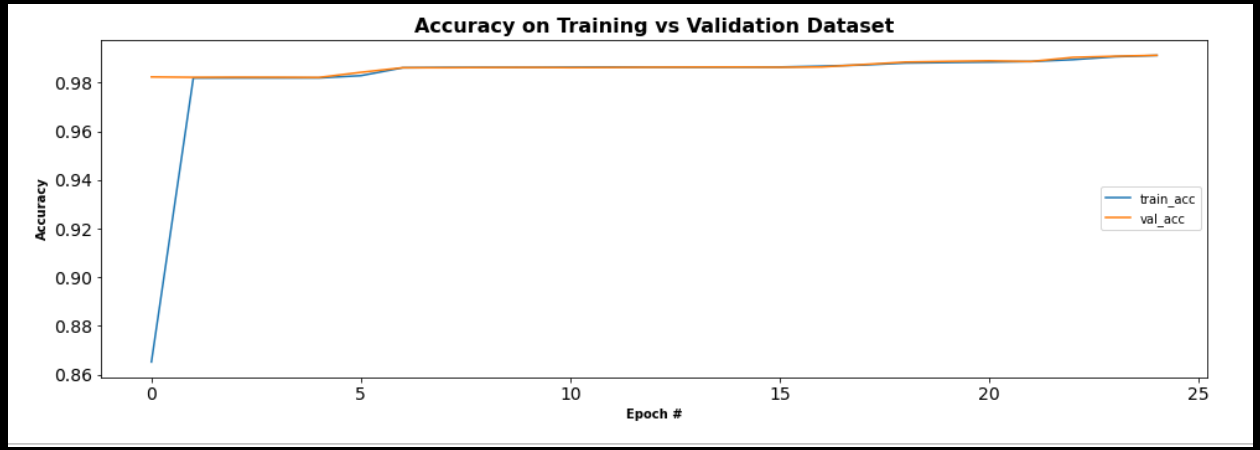
2nd Option: Removing Object Columns

Accuracy = 0.9840595111583422
Classification Report =

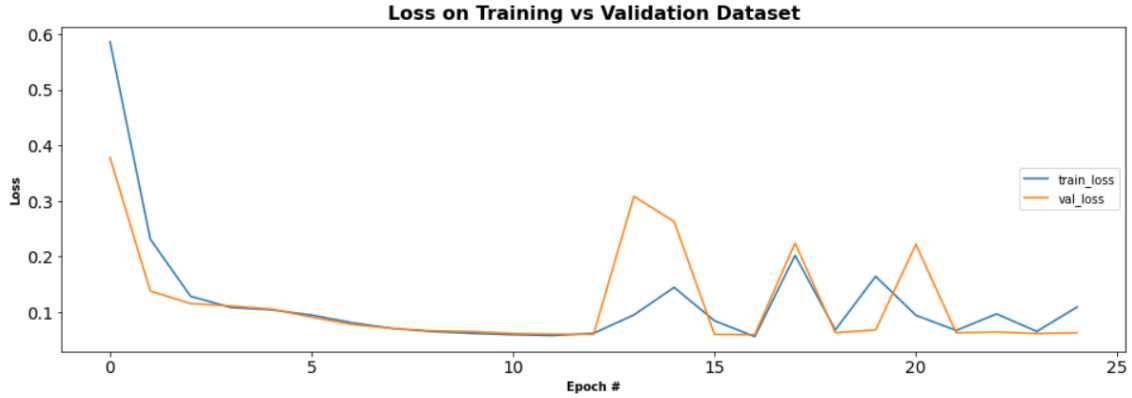
	precision	recall	f1-score	support
back.	0.05	0.02	0.03	443
buffer_overflow.	0.00	0.00	0.00	11
ftp_write.	0.00	0.00	0.00	1
guess_passwd.	0.00	0.00	0.00	13
imap.	0.00	0.00	0.00	2
ipsweep.	0.00	0.00	0.00	251
land.	0.00	0.00	0.00	2
loadmodule.	0.00	0.00	0.00	2
neptune.	1.00	1.00	1.00	21531
nmap.	0.00	0.00	0.00	52
normal.	0.98	0.99	0.99	19526
perl.	0.00	0.00	0.00	1
phf.	0.00	0.00	0.00	1
pod.	0.00	0.00	0.00	58
portsweep.	0.00	0.00	0.00	200
rootkit.	0.00	0.00	0.00	1
satan.	0.38	0.93	0.53	319
smurf.	0.99	1.00	1.00	55979
teardrop.	0.00	0.00	0.00	198
warezclient.	0.00	0.00	0.00	211
warezmaster.	0.00	0.00	0.00	3
accuracy			0.98	98805
macro avg	0.16	0.19	0.17	98805
weighted avg	0.98	0.98	0.98	98805

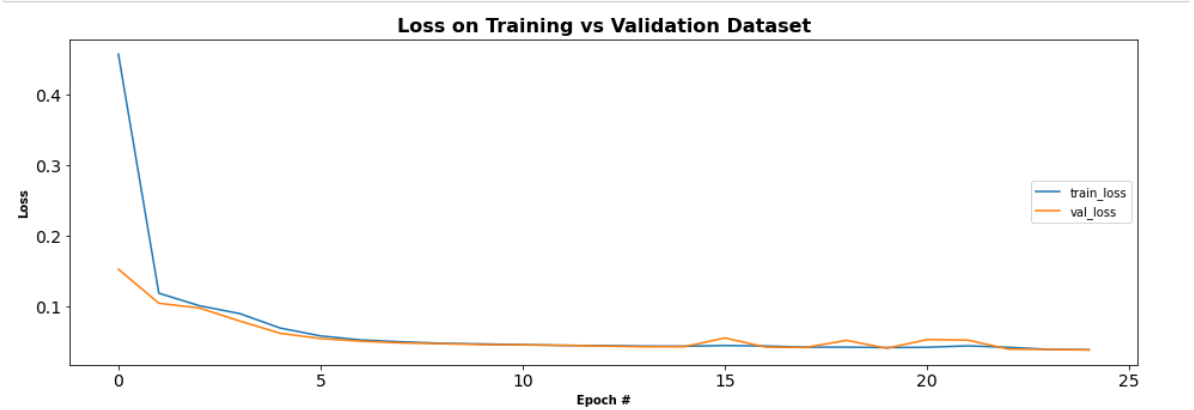
Neural Network

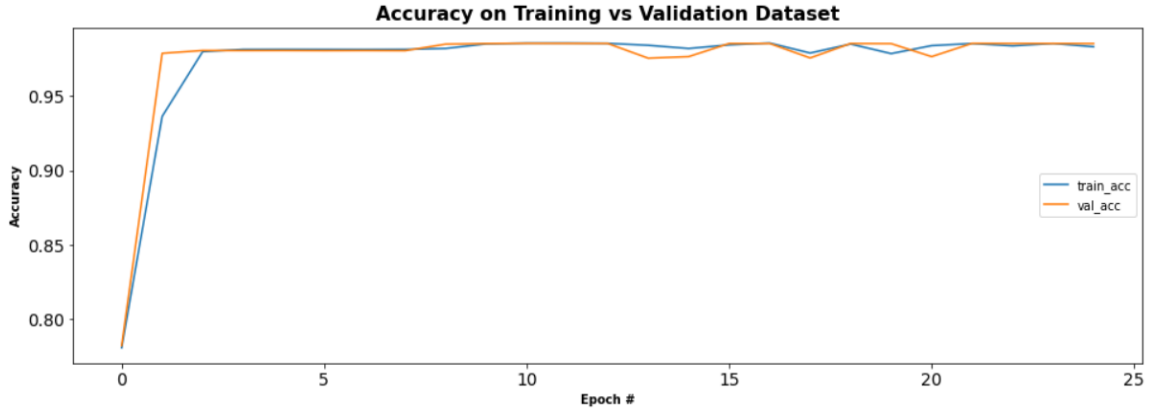


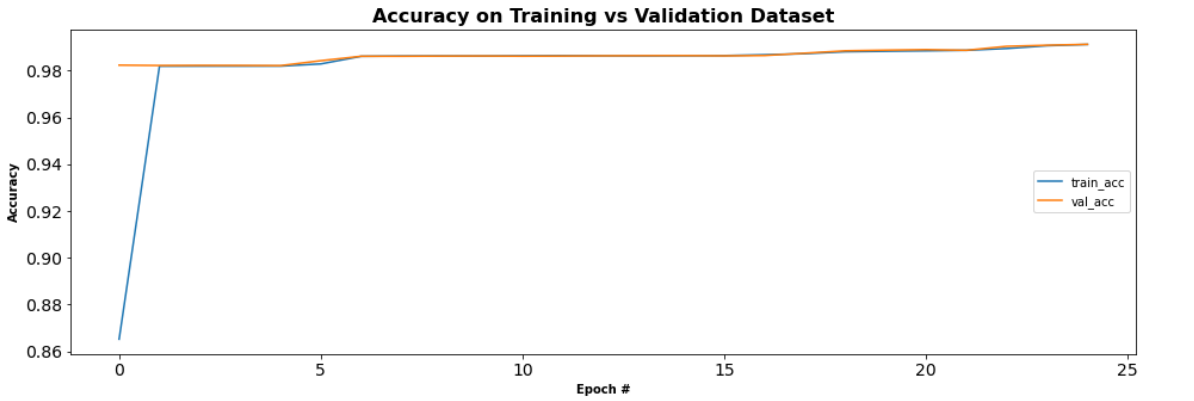


With Columns vs Without Columns









Logistic Regression

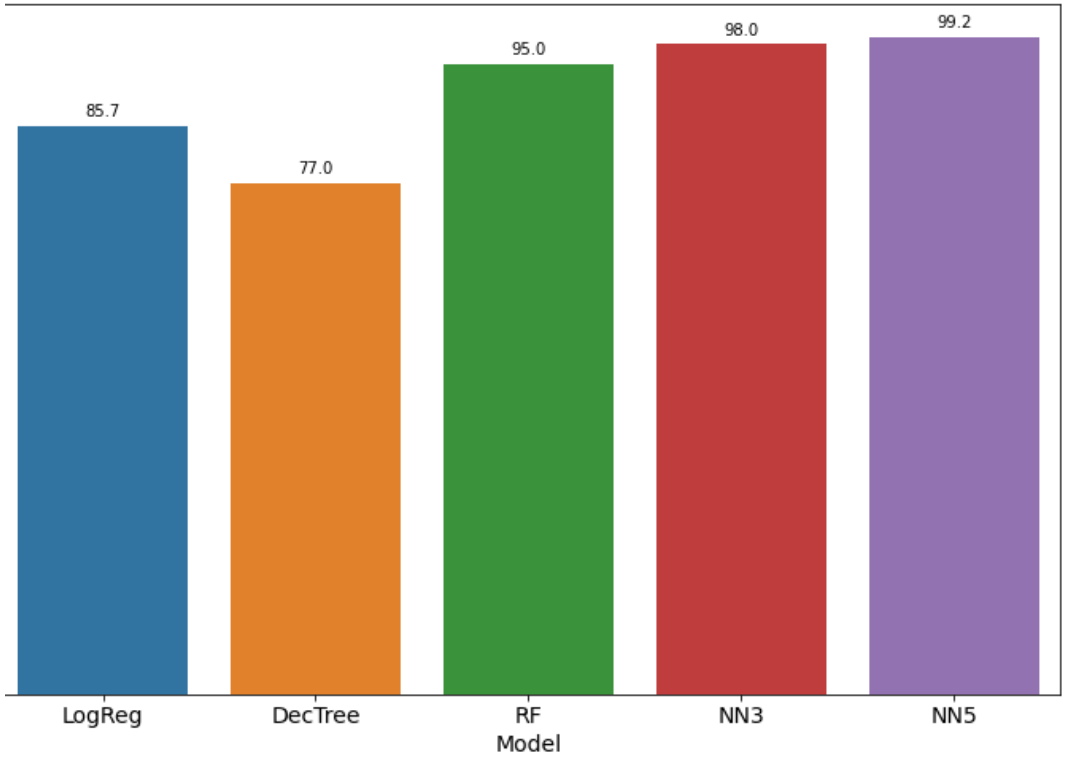
- Increasing the maximum number of iterations does not necessarily guarantee convergence.

```
C:\Users\Bryan\anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:764: ConvergenceWarning: lbfgs failed to converge
(status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
```

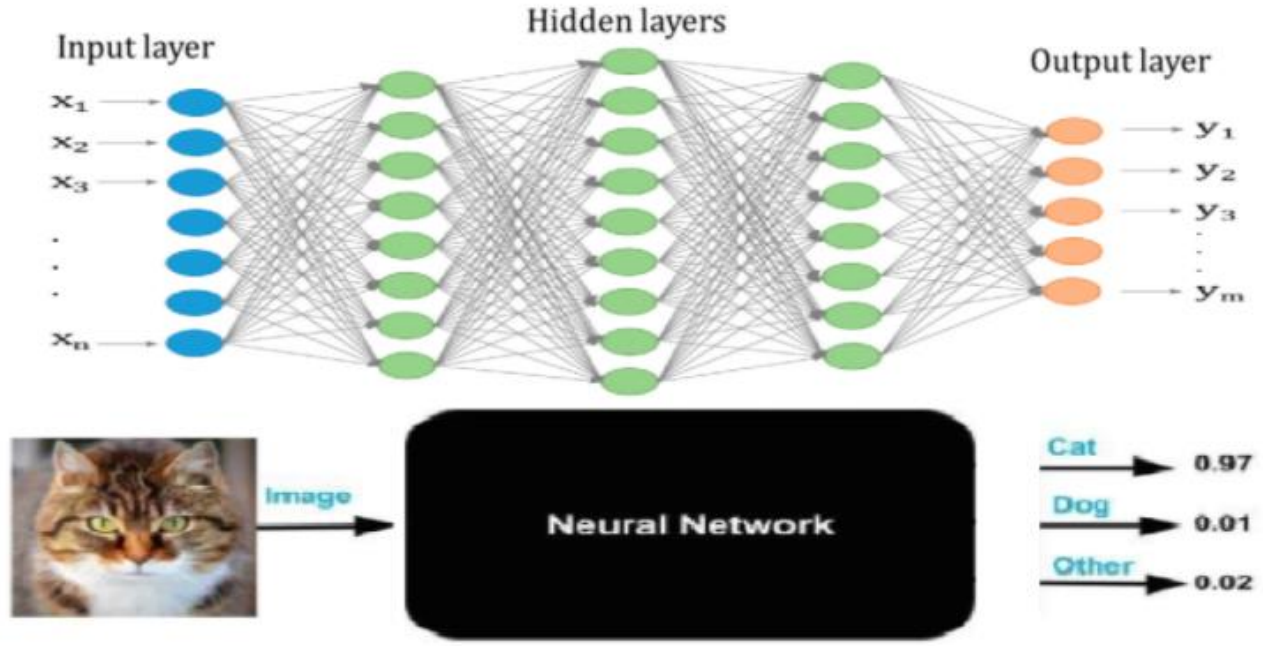
```
LogisticRegression(... solver='lbfgs', max_iter=100 ...)
```

Results

- Logistic Regression 85%
- Decision Tree 77%
- Random Forest 93-95%
- Neural Network 3 layer 97-98%
- Neural Network 5 layers 98-99%



What could be next ?



McDonald, Conor, Machine Learning Fundamentals. 29 December 2017 [7].

- Instead of using a prepared dataset, implement a way to use regular network traffic.
- Insert a type of traffic as input, and display the type of attack as an output

New Sources

- Yiu, T. (2019, August 14). Understanding Random Forest. Retrieved December 07, 2020, from https://towardsdatascience.com/understanding-random-forest-58381e0602d2
- Richer, V. (2019, March 04). Understanding Decision Trees (once and for all!) . Retrieved December 07, 2020, from https://towardsdatascience.com/understanding-decision-trees-once-and-for-all-2d891b1be579
- Liberman, N. (2020, May 21). Decision Trees and Random Forests. Retrieved December 07, 2020, from https://towardsdatascience.com/decision-trees-and-random-forests-df0c3123f991

Thank you!

Questions?