Grant Thornton

A Machine Learning Approach to Forensic Investigations (KNN)

Jashi Jeyaantony
IR Engineer

Problem/Model?

- Large datasets are time consuming to manually comb through, ML models can be applied for classification (i.e. Malicious, Not malicious)

- We will be looking at Windows event logs (4624 – Successful log on) as a PoC

- Model selection – K- Nearest neighbours
  - Supervised (labelled) ML model, Uses feature similarity for classification, Ignores data distribution patterns

Data

- Models are only as good as the data they are trained on.
- Collect and label data
  - Honeypot; naturally generates varied malicious activity
  - Pre-existing data sets; Labelled/Not Labelled
  - Simulation of attacks in a controlled Lab; highly specific

- My data: Simulated Active Directory lab, where lateral movement (pass-the-hash) and User and Device enumeration (Bloodhound)
- Raw data was extracted using EvtxECmd, Labelled with the help of APT

Feature Generation

- Time, Account name, Workstation name, Source IP
- Raw data -> Meaningful data
- Features of this data:
  - Islan: Is it a Lan IP?
  - Isnewip: Are there reoccurring logins in past 7 days from this IP?
  - Isvpn: Is it a VPN?
  - Precent: % of logins in the past day?
  - Src_ip_c: Successful login in 15min window?
  - Tag: Malicious or not?

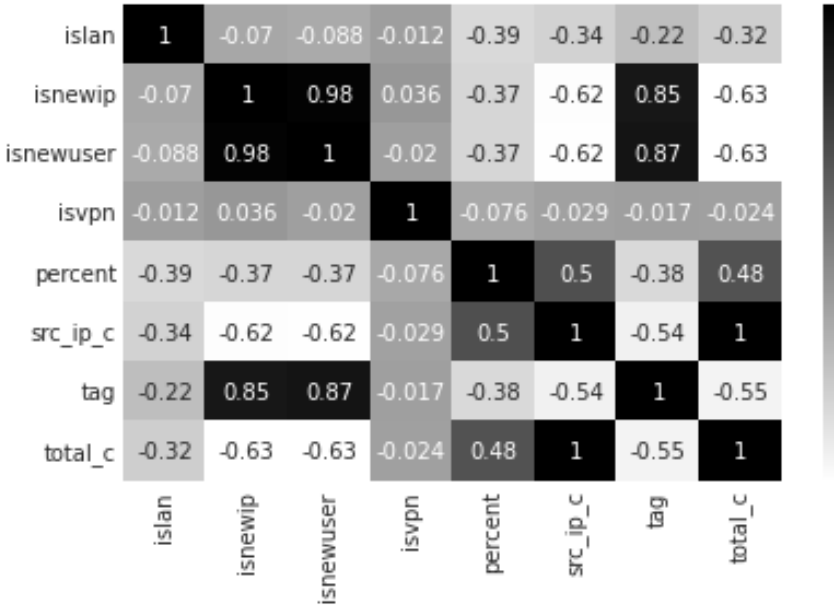
Code

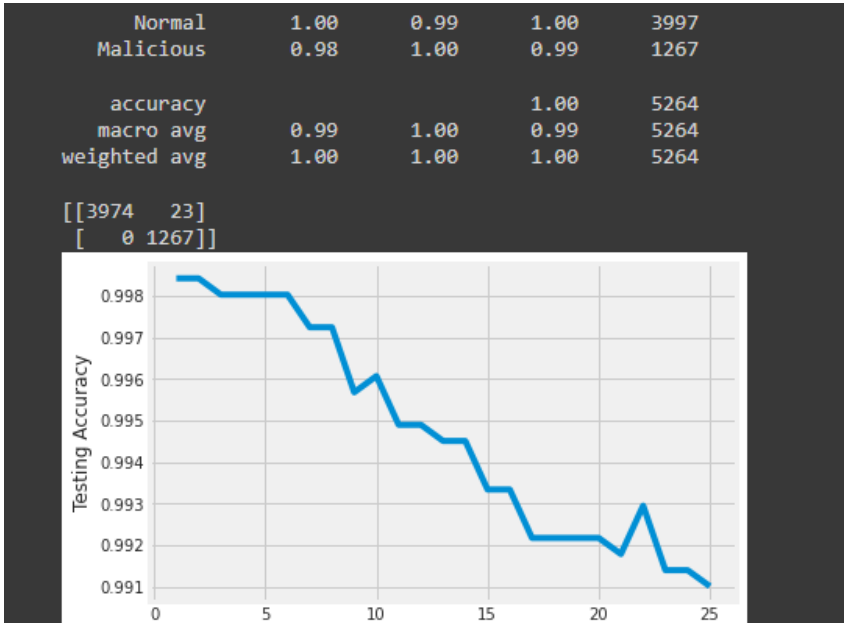
```
feature_cols = ['isnewuser','isnewip','isvpn','islan','percent','src_ip_c']
X = df[feature_cols]
y = df['tag']
# train and split data for testing
X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.4)
k_range= range(1,26)
scores = []
# Finding the best K
for k in k_range:
    knn = KNeighborsClassifier(n_neighbors=k)
    knn.fit(X_train, y_train)
    y_pred = knn.predict(X_test)
    scores.append(metrics.f1_score(y_test, y_pred))

    targetname = ['Normal','Malicious']
    result = metrics.classification_report(y_test,y_pred,target_names=targetname)
    matrix = metrics.confusion_matrix(y_test, y_pred, labels = [0,1])
    print ('Currently running K:' + str(k))
    print (result)
    print (matrix)
# Plotting as graph
plt.plot(k_range, scores)
plt.xlabel('Value of K for KNN')
plt.ylabel('Testing Accuracy')
plt.show()
```

- Models tested: Xgboost, KNN, ANN
- Snippet of the KNN model
- 60/40, train test split
- 25 epochs
- Code/data available on github

Grant Thornton

Training and Results





Grant Thornton

Future works

- Create more complex model (deep learning) with larger databases and more inputs.

  -> Leverage currently archived investigation data using investigation notes, kape, apt…etc to create the ***holy grail*** in AI models for IR

- Application of ML in SOC? – Open call for ideas
- Ongoing ML projects – Malware classification using tagging
- Thank you for your attention!

Grant Thornton