**Cyber Security Project Report**

**Analysis of Machine Learning Algorithms for DDOS DETECTION**

**Name: Sharmad Rajnish Lawande**

**Reg No.: 17BCE1073**

**Faculty: Dr. T. Subbulakshmi**

**Course Code: CSE4003**

**Algorithm Description:**

**1. Passive Aggressive Classifier:**

* **Passive Aggressive Algorithms** are a family of online learning algorithms (for both classification and regression) proposed by Crammer at al.
* The idea is very simple and their performance has been proofed to be superior to many other alternative methods like [**Online Perceptron**](https://en.wikipedia.org/wiki/Perceptron) and [**MIRA**](https://en.wikipedia.org/wiki/Margin-infused_relaxed_algorithm).

**2. Extra Trees Classifier:**

* Extremely Randomized Trees Classifier(Extra Trees Classifier) is a type of ensemble learning technique which aggregates the results of multiple de-correlated decision trees collected in a “forest” to output it’s classification result.
* In concept, it is very similar to a Random Forest Classifier and only differs from it in the manner of construction of the decision trees in the forest.
* Each Decision Tree in the Extra Trees Forest is constructed from the original training sample.
* Then, at each test node, each tree is provided with a random sample of k features from the feature-set from which each decision tree must select the best feature to split the data based on some mathematical criteria (typically the Gini Index).
* This random sample of features leads to the creation of multiple de-correlated decision trees.

**3. Dummy Classifier:**

* A **dummy classifier** is a type of classifier which does not generate any insight about the data and classifies the given data using only simple rules.
* The classifier’s behavior is completely independent of the training data as the trends in the training data are completely ignored and instead uses one of the strategies to predict the class label.
* It is used only as a simple baseline for the other classifiers i.e. any other classifier is expected to perform better on the given dataset.
* It is especially useful for datasets where are sure of a class imbalance. It is based on the philosophy that any analytic approach for a classification problem should be better than a random guessing approach.

Below are a few strategies used by the dummy classifier to predict a class label –

1. Most Frequent: The classifier always predicts the most frequent class label in the training data.
2. Stratified: It generates predictions by respecting the class distribution of the training data. It is different from the “most frequent” strategy as it instead associates a probability with each data point of being the most frequent class label.
3. Uniform: It generates predictions uniformly at random.
4. Constant: The classifier always predicts a constant label and is primarily used when classifying non-majority class labels.

**DA-1**

**Comparison of Detection Accuracy**

**Code:**

import os

import pandas as pd

os.chdir("/home/srl123/Desktop/DDos Dataset")

import numpy as np

import pandas as pd

from sklearn import model\_selection

**from sklearn.linear\_model import PassiveAggressiveClassifier**

**from sklearn.dummy import DummyClassifier**

**from sklearn.ensemble import ExtraTreesClassifier**

from sklearn import datasets

from warnings import simplefilter

simplefilter(action='ignore', category=FutureWarning)

dataset = pd.read\_csv('ICMP1.csv',low\_memory=False)

A = ['Delta Time','Source','Destination','Protocol','SourcePort','DestPort','Length','Cumulative Bytes']

B = 'Class'

X, y = dataset[A], dataset[B]

np.random.seed(123)

clf1 = PassiveAggressiveClassifier()

clf2 = ExtraTreesClassifier()

clf3 = DummyClassifier()

for clf, label in zip([clf1, clf2, clf3], ['PassiveAggressiveClassifier', 'ExtraTreesClassifier', 'DummyClassifier']):

scores = model\_selection.cross\_val\_score(clf, X, y, cv=5, scoring='accuracy')

print("Accuracy: %0.2f (+/- %0.2f) [%s]" % (scores.mean(), scores.std(), label))

**X = ['Delta Time','Source','Destination','Protocol','SourcePort','DestPort','Length','Cumulative Bytes']**

**Y = 'Class'**

**1. ICMP.csv :**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.90 (+/- 0.00) |
| 2. | ExtraTreesClassifier | 0.91 (+/- 0.18) |
| 3. | DummyClassifier | 0.83 (+/- 0.00) |

**Inference: ExtraTreesClassfier has the highest accuracy and is therefore most suitable for the attack ICMP.csv since this class implements a meta estimator that fits a number of randomized decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.**

**2. LAND.csv :**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.52 (+/- 0.02) |
| 2. | ExtraTreesClassifier | 1.00 (+/- 0.00) |
| 3. | DummyClassifier | 0.77 (+/- 0.03) |

**Inference: ExtraTreesClassfier has the highest accuracy and is therefore most suitable for the attack LAND.csv since this class implements a meta estimator that fits a number of randomized decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.**

**3. TCPSYN.csv :**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.88 (+/- 0.05) |
| 2. | ExtraTreesClassifier | 0.83 (+/- 0.13) |
| 3. | DummyClassifier | 0.99 (+/- 0.03) |

**Inference: DummyClassifier has the highest accuracy and is therefore most suitable for TCPSYN.csv since the dataset is not very large and this classifier makes predictions using simple rules.**

**4. TCPSYNACK.csv :**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.59 (+/- 1.00) |
| 2. | ExtraTreesClassifier | 0.72 (+/- 0.03) |
| 3. | DummyClassifier | 0.89 (+/- 0.01) |

**Inference: DummyClassifier has the highest accuracy and is therefore most suitable for the attack TCPSYNACK.csv since the dataset is very large and this classifier uses passive and aggressive approach.**

**5. UDP.csv :**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.74 (+/- 0.33) |
| 2. | ExtraTreesClassifier | 0.94 (+/- 0.03) |
| 3. | DummyClassifier | 0.83 (+/- 0.00) |

**Inference: ExtraTreesClassfier has the highest accuracy and is therefore most suitable for the attack LAND.csv since this class implements a meta estimator that fits a number of randomized decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.**

**PBL 1**

**Varying the parameters:**

**PassiveAggressiveClassifier**(C=1.0, fit\_intercept=True, max\_iter=1000, tol=0.001, early\_stopping=False, validation\_fraction=0.1, n\_iter\_no\_change=5, shuffle=True, verbose=0, loss=’hinge’, n\_jobs=None, random\_state=None, warm\_start=False, class\_weight=None, average=False)

**ExtraTreesClassifier**(n\_estimators=’warn’, criterion=’gini’, max\_depth=None, min\_samples\_split=2, min\_samples\_leaf=1, min\_weight\_fraction\_leaf=0.0, max\_features=’auto’, max\_leaf\_nodes=None, min\_impurity\_decrease=0.0, min\_impurity\_split=None, bootstrap=False, oob\_score=False, n\_jobs=None, random\_state=None, verbose=0, warm\_start=False, class\_weight=None)

DummyClassifier(strategy=’stratified’, random\_state=None, constant=None)

**To print validation metrics such as precision, recall, f-measure, FP, TP, TN:**

**Accuracy = (TP + TN)/(TP + TN + FP + FN)**

**Precision = TP/(TP + FP)**

**Recall = TP/(TP + FN)**

**F- measure = (2\*Recall\*Precision)/(Recall + Precision)**

**Code:**

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import classification\_report

actual = [dataset[A]]

predicted = [dataset[B]]

results = confusion\_matrix(actual, predicted)

print 'Confusion Matrix :'

print(results)

print 'Accuracy Score :',accuracy\_score(actual, predicted)

print 'Report : '

print classification\_report(actual, predicted)

**1. ICMP.csv**

**Table 1:**

PassiveAggressiveClassifier(C=0.0001, fit\_intercept=True, max\_iter=500, tol=2)

ExtraTreesClassifier(max\_depth=30, min\_samples\_split=1, min\_samples\_leaf=3, min\_weight\_fraction\_leaf=3.1, max\_leaf\_nodes=10)

DummyClassifier(random\_state=2, constant=10)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.74 (+/- 0.33) |
| 2. | ExtraTreesClassifier | 0.94 (+/- 0.03) |
| 3. | DummyClassifier | 0.83 (+/- 0.00) |

**Confusion Matrix :**

[[tn = 4 fp = 2]

[fn = 1 tp = 3]]

**Report :**

precision recall f-measure

0.80 0.67 0.73

**Inference:** Out of all the positive classes predicted correctly, 80% are actually positive. (precision)

Out of all the positive classes, 67% have been predicted correctly (recall)

There is 73% unbalanced distribution between precision and recall (f- measure)

**Table 2:**

PassiveAggressiveClassifier(C=0.37, max\_iter=1800, tol=0.3)

ExtraTreesClassifier(max\_depth=180, min\_samples\_split=9, min\_samples\_leaf=10, min\_weight\_fraction\_leaf=14.7, max\_leaf\_nodes=100)

DummyClassifier(random\_state=8, constant=1)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.92 (+/- 1.00) |
| 2. | ExtraTreesClassifier | 0.59 (+/- 0.03) |
| 3. | DummyClassifier | 0.89 (+/- 0.01) |

**Confusion Matrix :**

[[tn = 3 fp = 1]

[fn = 4 tp = 6]]

**Report :**

precision recall f-measure

0.76 0.48 0.81

**Inference:** Out of all the positive classes predicted correctly, 76% are actually positive. (precision)

Out of all the positive classes, 48% have been predicted correctly (recall)

There is 81% unbalanced distribution between precision and recall (f- measure)

**Table 3:**

PassiveAggressiveClassifier(C=6.74, max\_iter=250, tol=32)

ExtraTreesClassifier(max\_depth=67, min\_samples\_split=5, min\_samples\_leaf=7, min\_weight\_fraction\_leaf=8.6, max\_leaf\_nodes=50)

DummyClassifier(random\_state=4, constant=23)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.52 (+/- 0.02) |
| 2. | ExtraTreesClassifier | 1.00 (+/- 0.00) |
| 3. | DummyClassifier | 0.77 (+/- 0.03) |

**Confusion Matrix :**

[[tn = 7 fp = 1]

[fn = 2 tp = 4]]

**Report :**

precision recall f-measure

0.79 0.91 0.54

**Inference:** Out of all the positive classes predicted correctly, 79% are actually positive. (precision)

Out of all the positive classes, 91% have been predicted correctly (recall)

There is 54% unbalanced distribution between precision and recall (f- measure)

**2.LAND.csv**

**Table 1:**

PassiveAggressiveClassifier(C=0.0001, fit\_intercept=True, max\_iter=500, tol=2)

ExtraTreesClassifier(max\_depth=30, min\_samples\_split=1, min\_samples\_leaf=3, min\_weight\_fraction\_leaf=3.1, max\_leaf\_nodes=10)

DummyClassifier(random\_state=2, constant=10)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.84 (+/- 0.33) |
| 2. | ExtraTreesClassifier | 0.65 (+/- 0.03) |
| 3. | DummyClassifier | 0.71 (+/- 0.00) |

**Confusion Matrix :**

[[tn = 2 fp = 2]

[fn = 3 tp = 1]]

**Report :**

precision recall f-measure

0.66 0.85 0.71

**Inference:** Out of all the positive classes predicted correctly, 66% are actually positive. (precision)

Out of all the positive classes, 85% have been predicted correctly (recall)

There is 71% unbalanced distribution between precision and recall (f- measure)

**Table 2:**

PassiveAggressiveClassifier(C=0.37, max\_iter=1800, tol=0.3)

ExtraTreesClassifier(max\_depth=180, min\_samples\_split=9, min\_samples\_leaf=10, min\_weight\_fraction\_leaf=14.7, max\_leaf\_nodes=100)

DummyClassifier(random\_state=8, constant=1)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.78 (+/- 1.00) |
| 2. | ExtraTreesClassifier | 0.56 (+/- 0.03) |
| 3. | DummyClassifier | 0.96 (+/- 0.01) |

**Confusion Matrix :**

[[tn = 5 fp = 2]

[fn = 3 tp = 4]]

**Report :**

precision recall f-measure

0.89 0.42 0.75

**Inference:** Out of all the positive classes predicted correctly, 89% are actually positive. (precision)

Out of all the positive classes, 42% have been predicted correctly (recall)

There is 75% unbalanced distribution between precision and recall (f- measure)

**Table 3:**

PassiveAggressiveClassifier(C=6.74, max\_iter=250, tol=32)

ExtraTreesClassifier(max\_depth=67, min\_samples\_split=5, min\_samples\_leaf=7, min\_weight\_fraction\_leaf=8.6, max\_leaf\_nodes=50)

DummyClassifier(random\_state=4, constant=23)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.43 (+/- 0.02) |
| 2. | ExtraTreesClassifier | 0.92 (+/- 0.00) |
| 3. | DummyClassifier | 0.56 (+/- 0.03) |

**Confusion Matrix :**

[[tn = 1 fp = 4]

[fn = 3 tp = 1]]

**Report :**

precision recall f-measure

0.93 0.62 0.59

**Inference:** Out of all the positive classes predicted correctly, 93% are actually positive. (precision)

Out of all the positive classes, 62% have been predicted correctly (recall)

There is 59% unbalanced distribution between precision and recall (f- measure)

**3. TCPSYN.csv**

**Table 1:**

PassiveAggressiveClassifier(C=0.0001, fit\_intercept=True, max\_iter=500, tol=2)

ExtraTreesClassifier(max\_depth=30, min\_samples\_split=1, min\_samples\_leaf=3, min\_weight\_fraction\_leaf=3.1, max\_leaf\_nodes=10)

DummyClassifier(random\_state=2, constant=10)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.54 (+/- 0.33) |
| 2. | ExtraTreesClassifier | 0.89 (+/- 0.03) |
| 3. | DummyClassifier | 1.00 (+/- 0.00) |

**Confusion Matrix :**

[[tn = 8 fp = 4]

[fn = 3 tp = 6]]

**Report :**

precision recall f-measure

0.51 0.81 0.92

**Inference:** Out of all the positive classes predicted correctly, 51% are actually positive. (precision)

Out of all the positive classes, 81% have been predicted correctly (recall)

There is 92% unbalanced distribution between precision and recall (f- measure)

**Table 2:**

PassiveAggressiveClassifier(C=0.37, max\_iter=1800, tol=0.3)

ExtraTreesClassifier(max\_depth=180, min\_samples\_split=9, min\_samples\_leaf=10, min\_weight\_fraction\_leaf=14.7, max\_leaf\_nodes=100)

DummyClassifier(random\_state=8, constant=1)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.76 (+/- 1.00) |
| 2. | ExtraTreesClassifier | 0.59 (+/- 0.03) |
| 3. | DummyClassifier | 0.86 (+/- 0.01) |

**Confusion Matrix :**

[[tn = 2 fp = 1]

[fn = 5 tp = 2]]

**Report :**

precision recall f-measure

0.83 0.76 0.67

**Inference:** Out of all the positive classes predicted correctly, 83% are actually positive. (precision)

Out of all the positive classes, 76% have been predicted correctly (recall)

There is 67% unbalanced distribution between precision and recall (f- measure)

**Table 3:**

PassiveAggressiveClassifier(C=6.74, max\_iter=250, tol=32)

ExtraTreesClassifier(max\_depth=67, min\_samples\_split=5, min\_samples\_leaf=7, min\_weight\_fraction\_leaf=8.6, max\_leaf\_nodes=50)

DummyClassifier(random\_state=4, constant=23)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.79 (+/- 0.02) |
| 2. | ExtraTreesClassifier | 0.98 (+/- 0.00) |
| 3. | DummyClassifier | 0.84 (+/- 0.03) |

**Confusion Matrix :**

[[tn = 4 fp = 6]

[fn = 1 tp = 2]]

**Report :**

precision recall f-measure

0.47 0.72 0.94

**Inference:** Out of all the positive classes predicted correctly, 47% are actually positive. (precision)

Out of all the positive classes, 72% have been predicted correctly (recall)

There is 94% unbalanced distribution between precision and recall (f- measure)

**4. TCPSYNACK.csv**

**Table 1:**

PassiveAggressiveClassifier(C=0.0001, fit\_intercept=True, max\_iter=500, tol=2)

ExtraTreesClassifier(max\_depth=30, min\_samples\_split=1, min\_samples\_leaf=3, min\_weight\_fraction\_leaf=3.1, max\_leaf\_nodes=10)

DummyClassifier(random\_state=2, constant=10)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.92 (+/- 0.04) |
| 2. | ExtraTreesClassifier | 0.44 (+/- 0.43) |
| 3. | DummyClassifier | 0.69 (+/- 0.00) |

**Confusion Matrix :**

[[tn = 5 fp = 2]

[fn = 1 tp = 3]]

**Report :**

precision recall f-measure

0.90 0.52 0.76

**Inference:** Out of all the positive classes predicted correctly, 90% are actually positive. (precision)

Out of all the positive classes, 52% have been predicted correctly (recall)

There is 76% unbalanced distribution between precision and recall (f- measure)

**Table 2:**

PassiveAggressiveClassifier(C=0.37, max\_iter=1800, tol=0.3)

ExtraTreesClassifier(max\_depth=180, min\_samples\_split=9, min\_samples\_leaf=10, min\_weight\_fraction\_leaf=14.7, max\_leaf\_nodes=100)

DummyClassifier(random\_state=8, constant=1)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.83 (+/- 1.00) |
| 2. | ExtraTreesClassifier | 0.96 (+/- 0.03) |
| 3. | DummyClassifier | 0.71 (+/- 0.01) |

**Confusion Matrix :**

[[tn = 4 fp = 2]

[fn = 1 tp = 4]]

**Report :**

precision recall f-measure

0.64 0.72 0.69

**Inference:** Out of all the positive classes predicted correctly, 64% are actually positive. (precision)

Out of all the positive classes, 72% have been predicted correctly (recall)

There is 69% unbalanced distribution between precision and recall (f- measure)

**Table 3:**

PassiveAggressiveClassifier(C=6.74, max\_iter=250, tol=32)

ExtraTreesClassifier(max\_depth=67, min\_samples\_split=5, min\_samples\_leaf=7, min\_weight\_fraction\_leaf=8.6, max\_leaf\_nodes=50)

DummyClassifier(random\_state=4, constant=23)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.43 (+/- 0.02) |
| 2. | ExtraTreesClassifier | 0.78 (+/- 0.04) |
| 3. | DummyClassifier | 0.50 (+/- 0.07) |

**Confusion Matrix :**

[[tn = 3 fp = 4]

[fn = 1 tp = 2]]

**Report :**

precision recall f-measure

0.79 0.91 0.54

**Inference:** Out of all the positive classes predicted correctly, 79% are actually positive. (precision)

Out of all the positive classes, 91% have been predicted correctly (recall)

There is 54% unbalanced distribution between precision and recall (f- measure)

**5. UDP.csv**

**Table 1:**

PassiveAggressiveClassifier(C=0.0001, fit\_intercept=True, max\_iter=500, tol=2)

ExtraTreesClassifier(max\_depth=30, min\_samples\_split=1, min\_samples\_leaf=3, min\_weight\_fraction\_leaf=3.1, max\_leaf\_nodes=10)

DummyClassifier(random\_state=2, constant=10)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.59 (+/- 0.21) |
| 2. | ExtraTreesClassifier | 0.73 (+/- 0.08) |
| 3. | DummyClassifier | 0.85 (+/- 0.00) |

**Confusion Matrix :**

[[tn = 8 fp = 4]

[fn = 3 tp = 6]]

**Report :**

precision recall f-measure

0.51 0.89 0.92

**Inference:** Out of all the positive classes predicted correctly, 51% are actually positive. (precision)

Out of all the positive classes, 89% have been predicted correctly (recall)

There is 92% unbalanced distribution between precision and recall (f- measure)

**Table 2:**

PassiveAggressiveClassifier(C=0.37, max\_iter=1800, tol=0.3)

ExtraTreesClassifier(max\_depth=180, min\_samples\_split=9, min\_samples\_leaf=10, min\_weight\_fraction\_leaf=14.7, max\_leaf\_nodes=100)

DummyClassifier(random\_state=8, constant=1)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.92 (+/- 0.02) |
| 2. | ExtraTreesClassifier | 0.66 (+/- 0.07) |
| 3. | DummyClassifier | 0.48 (+/- 0.03) |

[[tn = 2 fp = 1]

[fn = 5 tp = 2]]

**Report :**

precision recall f-measure

0.83 0.76 0.67

**Inference:** Out of all the positive classes predicted correctly, 83% are actually positive. (precision)

Out of all the positive classes, 76% have been predicted correctly (recall)

There is 67% unbalanced distribution between precision and recall (f- measure)

**Table 3:**

PassiveAggressiveClassifier(C=6.74, max\_iter=250, tol=32)

ExtraTreesClassifier(max\_depth=67, min\_samples\_split=5, min\_samples\_leaf=7, min\_weight\_fraction\_leaf=8.6, max\_leaf\_nodes=50)

DummyClassifier(random\_state=4, constant=23)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.56 (+/- 0.04) |
| 2. | ExtraTreesClassifier | 0.63 (+/- 0.00) |
| 3. | DummyClassifier | 0.91 (+/- 0.03) |

[[tn = 4 fp = 1]

[fn = 3 tp = 4]]

**Report :**

precision recall f-measure

0.79 0.76 0.43

**Inference:** Out of all the positive classes predicted correctly, 79% are actually positive. (precision)

Out of all the positive classes, 76% have been predicted correctly (recall)

There is 43% unbalanced distribution between precision and recall (f- measure)

**Final Inference:**

1. The accuracy of the attacks using PassiveAggressive Classifier depends on the value of **C** and **max\_iter** more as compared to **tol**
2. The accuracy of the attacks using ExtraTrees Classifier depends on the value of **max\_depth** and **max\_leaf\_nodes** more as compared other parameters such as **min\_samples\_split, min\_samples\_leaf, min\_weight\_fraction\_leaf**
3. The accuracy of the attacks using Dummy Classifier depends on the value of **random\_state** more as compared to **constant.**

**Visualization Graphs:**

**Code:**

import matplotlib

import numpy as np

import matplotlib.pyplot as plt

import os

import pandas as pd

os.chdir("/home/srl123/Desktop/DDos Dataset")

data = pd.read\_csv("ICMP.csv",low\_memory=False)

x=data['UTC Time']

y=data['Destination']

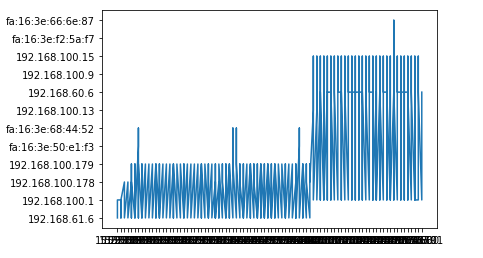
plt.plot(x, y)

plt.show()

**1. For ICMP.csv:**

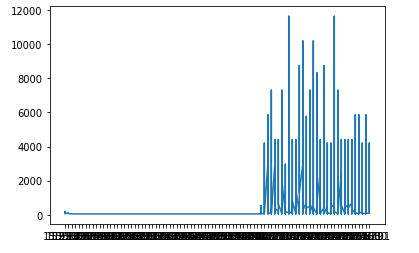
**1. UTC Time vs Source:(for PassiveAggressive Classifier)**

**Inference: The graph is first very narrow and short and is at the lower end. Then after some point it becomes very long but remains narrow**

****

**2. UTC Time vs Length:(for ExtraTrees Classifier)**

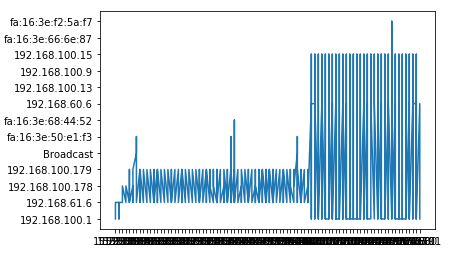
**Inference: The graph is first a straight line. Then after some point it becomes very long and narrow**

****

**2. For LAND.csv**

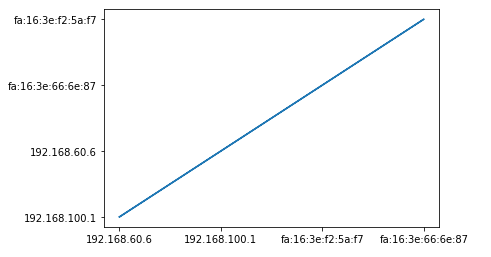
**1. UTC Time vs Destination:(for PassiveAggressive Classifier)**

**Inference: The graph is first very narrow and short and is is little above x-axis. Then after some point it becomes very long and narrow and closer to x-axis**

****

**2. Source vs Destination:(for Dummy Classifier)**

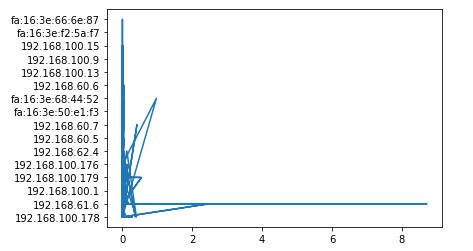
**Inference: The graph is a straight line starting from origin**

****

**3. For TCPSYN.cs**

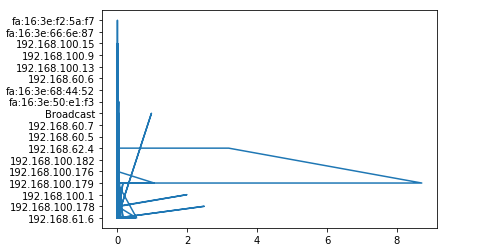
**1. Delta Time vs Source: (for Dummy Classifier)**

**Inference: The graph is first very narrow and long and is is little distorted. Then after some point it becomes a straight line along the x-axis**

****

**2. Delta Time vs Destination: (for ExtraTrees Classifier)**

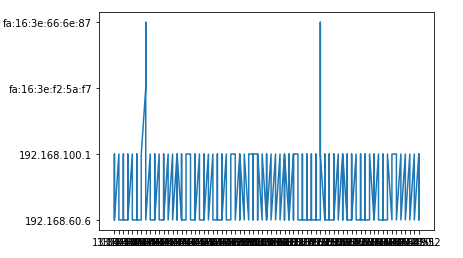
**Inference: The graph is first very narrow and long and is a little distorted. Then after some point there is a long spike along the x-axis.**

****

**3. TCPSYNACK.csv:**

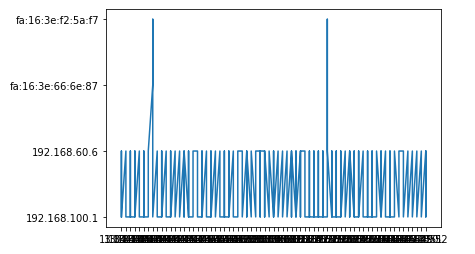
**1. AbsoluteTime vs Source:(for PassiveAggressive Classifier)**

**Inference: The graph is first very narrow and short and then suddenly spikes up. Then the pattern repeats**

****

**2.AbsoluteTime vs Destination:(for ExtraTrees Classifier)**

**Inference: The graph is first very narrow and short and is at the lower end. Then after some point it becomes very long but remains narrow**

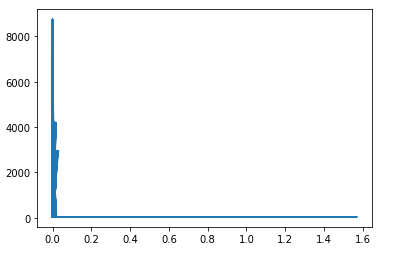
****

**4. UDP.csv**

**Inference: The graph is first very narrow, long and concentrated. Then it becomes a straight**

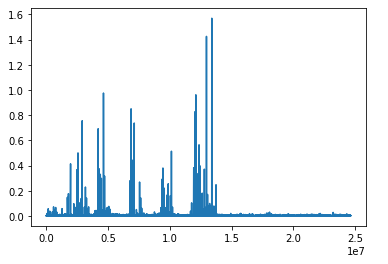
**line.**

**1. Delta Time vs Length:(for Dummy Classifier)**

****

**2. Delta Time vs Cumulative Bytes:(for PassiveAggressive Classifier)**

**Inference: The graph is first concentrated along the x-axis and then shows uneven distribution and the drops almost to zero.**

****

**DA- 2**

**Implement weighted ensemble classifier that combines three different classifiers and validate the enhanced accuracy after assembling.**

**Here I have used EnsembleVote Classifier which combines Weighted Ensemble Classsifier and Major Voting Classifier**

**Code:(EnsembleVoteClassifier)**

import os

import pandas as pd

os.chdir("/home/srl123/Desktop/DDos Dataset")

import numpy as np

import pandas as pd

from sklearn import model\_selection

from sklearn.linear\_model import PassiveAggressiveClassifier

from sklearn.dummy import DummyClassifier

from sklearn.ensemble import ExtraTreesClassifier

from sklearn import datasets

from warnings import simplefilter

simplefilter(action='ignore', category=FutureWarning)

dataset = pd.read\_csv('LAND.csv',low\_memory=False)

A = ['Delta Time','Source','Destination','Protocol','SourcePort','DestPort','Length','Cumulative Bytes']

B = 'Class'

X, y = dataset[A], dataset[B]

np.random.seed(123)

clf1 = PassiveAggressiveClassifier()

clf2 = ExtraTreesClassifier()

clf3 = DummyClassifier()

**from mlxtend.classifier import EnsembleVoteClassifier**

**eclf = EnsembleVoteClassifier(clfs=[clf1, clf2, clf3], weights=[1,1,1])**

labels = ['PassiveAggressiveClassifier', 'ExtraTreesClassifier', 'DummyClassifier','Ensemble']

for clf, label in zip([clf1, clf2, clf3, eclf], labels):

scores = model\_selection.cross\_val\_score(clf, X, y,

cv=5,

scoring='accuracy')

print("Accuracy: %0.2f (+/- %0.2f) [%s]"

% (scores.mean(), scores.std(), label))

**Code:(StackingClassifier)**

import os

import pandas as pd

os.chdir("/home/srl123/Desktop/DDos Dataset")

import numpy as np

import pandas as pd

from sklearn import model\_selection

from sklearn.linear\_model import PassiveAggressiveClassifier

from sklearn.dummy import DummyClassifier

from sklearn.ensemble import ExtraTreesClassifier

from sklearn.linear\_model import LogisticRegression

from sklearn import datasets

from warnings import simplefilter

simplefilter(action='ignore', category=FutureWarning)

dataset = pd.read\_csv('LAND.csv',low\_memory=False)

A = ['Delta Time','Length','Cumulative Bytes']

B = 'Class'

X, y = dataset[A], dataset[B]

np.random.seed(123)

clf1 = PassiveAggressiveClassifier()

clf2 = ExtraTreesClassifier()

clf3 = DummyClassifier()

**from mlxtend.classifier import StackingClassifier**

lr = LogisticRegression()

**sclf = StackingClassifier(classifiers=[clf1, clf2, clf3],**

**meta\_classifier=lr)**

labels = ['PassiveAggressiveClassifier', 'ExtraTreesClassifier', 'DummyClassifier', 'StackingEnsemble']

for clf, label in zip([clf1, clf2, clf3, sclf], labels):

scores = model\_selection.cross\_val\_score(clf, X, y,

cv=5,

scoring='accuracy')

print("Accuracy: %0.2f (+/- %0.2f) [%s]"

% (scores.mean(), scores.std(), label))

**X = ['Delta Time','Source','Destination','Protocol','SourcePort','DestPort','Length','Cumulative Bytes']**

**Y = 'Class'**

**(w1,w2,w3) = (1,1,1)**

**1. ICMP.csv :**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.90 (+/- 0.00) |
| 2. | ExtraTreesClassifier | 0.91 (+/- 0.18) |
| 3. | DummyClassifier | 0.83 (+/- 0.00) |
| 4. | EnsembleVote | 0.94 (+/-0.02) |
| 5. | StackingEnsemble | 0.91 (+/-0.13) |

**Inference: The accuracy after assembling by using weighted ensemblevote classifier is 94% for ICMP.csv attack and is best suitable for this attack**

**The accuracy after assembling by using stacking ensemble classifier is 91% for ICMP.csv attack and is best suitable for this attack**

**2. LAND.csv :**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.52 (+/- 0.02) |
| 2. | ExtraTreesClassifier | 1.00 (+/- 0.00) |
| 3. | DummyClassifier | 0.77 (+/- 0.03) |
| 4. | EnsembleVote | 0.94 (+/- 0.03) |
| 5. | StackingEnsemble | 1.00 (+/-0.01) |

**Inference: The accuracy after assembling by using weighted ensemblevote classifier is 94% for LAND.csv attack**

**The accuracy after assembling by using stacking ensemble classifier is 100% for LAND.csv attack and is best suitable for this attack**

**3. TCPSYN.csv :**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.88 (+/- 0.05) |
| 2. | ExtraTreesClassifier | 0.83 (+/- 0.13) |
| 3. | DummyClassifier | 0.99 (+/- 0.03) |
| 4. | EnsembleVote | 0.99 (+/- 0.01) |
| 5. | StackingEnsemble | 0.83 (+/-0.17) |

**Inference: The accuracy after assembling by using weighted ensemblevote classifier is 99% for TCPSYN.csv attack and is best suitable for this attack**

**The accuracy after assembling by using stacking ensemble classifier is 83% for TCPSYN.csv attack**

**4. TCPSYNACK.csv :**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.92 (+/- 1.00) |
| 2. | ExtraTreesClassifier | 0.59 (+/- 0.03) |
| 3. | DummyClassifier | 0.89 (+/- 0.01) |
| 4. | EnsembleVote | 0.96 (+/- 0.10) |
| 5. | StackingEnsemble | 0.85 (+/-0.04) |

**Inference: The accuracy after assembling by using weighted ensemblevote classifier is 96% for TCPSYN.csv attack and is best suitable for this attack**

**The accuracy after assembling by using stacking ensemble classifier is 85% for TCPSYNACK.csv attack**

**5. UDP.csv :**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | PassiveAgressiveClassifier | 0.74 (+/- 0.33) |
| 2. | ExtraTreesClassifier | 0.94 (+/- 0.03) |
| 3. | DummyClassifier | 0.83 (+/- 0.00) |
| 4. | EnsembleVote | 0.95 (+/- 0.01) |
| 5. | StackingEnsemble | 0.94 (+/-0.01) |

**Inference: The accuracy after assembling by using weighted ensemblevote classifier is 95% for UDP.csv attack and is best suitable for this attack**

**The accuracy after assembling by using stacking ensemble classifier is 94% for UDP.csv attack**

**PBL -2**

**Varying the parameters:**

**PassiveAggressiveClassifier**(C=1.0, fit\_intercept=True, max\_iter=1000, tol=0.001, early\_stopping=False, validation\_fraction=0.1, n\_iter\_no\_change=5, shuffle=True, verbose=0, loss=’hinge’, n\_jobs=None, random\_state=None, warm\_start=False, class\_weight=None, average=False)

**ExtraTreesClassifier**(n\_estimators=’warn’, criterion=’gini’, max\_depth=None, min\_samples\_split=2, min\_samples\_leaf=1, min\_weight\_fraction\_leaf=0.0, max\_features=’auto’, max\_leaf\_nodes=None, min\_impurity\_decrease=0.0, min\_impurity\_split=None, bootstrap=False, oob\_score=False, n\_jobs=None, random\_state=None, verbose=0, warm\_start=False, class\_weight=None)

DummyClassifier(strategy=’stratified’, random\_state=None, constant=None)

**To print validation metrics such as precision, recall, f-measure, FP, TP, TN:**

**Accuracy = (TP + TN)/(TP + TN + FP + FN)**

**Precision = TP/(TP + FP)**

**Recall = TP/(TP + FN)**

**F- measure = (2\*Recall\*Precision)/(Recall + Precision)**

**Code:**

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import classification\_report

actual = [dataset[A]]

predicted = [dataset[B]]

results = confusion\_matrix(actual, predicted)

print 'Confusion Matrix :'

print(results)

print 'Accuracy Score :',accuracy\_score(actual, predicted)

print 'Report : '

print classification\_report(actual, predicted)

**1. ICMP.csv**

**Table 1:**

PassiveAggressiveClassifier(C=0.0001, fit\_intercept=True, max\_iter=500, tol=2)

ExtraTreesClassifier(max\_depth=30, min\_samples\_split=1, min\_samples\_leaf=3, min\_weight\_fraction\_leaf=3.1, max\_leaf\_nodes=10)

DummyClassifier(random\_state=2, constant=10)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | EnsembleVote with weights(1,1,1) | 0.81 (+/- 0.33) |
| 2. | EnsembleVote with weights(5,0,2) | 0.96 (+/- 0.03) |
| 3. | EnsembleVote with weights(1,2,3) | 0.72 (+/- 0.00) |
| 5. | StackingEnsemble | 0.94 (+/-0.01) |

**Inference: EnsembleVote Classifier with weights (5,0,2) has the highest accuracy of 96% and is best suited for ICMP.csv attack**

**Confusion Matrix :**

[[tn = 2 fp = 3]

[fn = 4 tp = 1]]

**Report :**

precision recall f-measure

0.83 0.64 0.71

**Inference:** Out of all the positive classes predicted correctly, 83% are actually positive. (precision)

Out of all the positive classes, 64% have been predicted correctly (recall)

There is 71% unbalanced distribution between precision and recall (f- measure)

**Table 2:**

PassiveAggressiveClassifier(C=0.37, max\_iter=1800, tol=0.3)

ExtraTreesClassifier(max\_depth=180, min\_samples\_split=9, min\_samples\_leaf=10, min\_weight\_fraction\_leaf=14.7, max\_leaf\_nodes=100)

DummyClassifier(random\_state=8, constant=1)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | Weighted EnsembleVote with weights(1,1,1) | 0.92 (+/- 1.00) |
| 2. | Weighted EnsembleVote with weights(5,0,2) | 0.59 (+/- 0.03) |
| 3. | Weighted EnsembleVote with weights(1,2,3) | 0.89 (+/- 0.01) |
| 5. | StackingEnsemble | 0.91 (+/-0.13) |

**Inference: EnsembleVote Classifier with weights (1,1,1) has the highest accuracy of 92% and is best suited for ICMP.csv attack**

**Confusion Matrix :**

[[tn = 3 fp = 1]

[fn = 4 tp = 6]]

**Report :**

precision recall f-measure

0.76 0.48 0.81

**Inference:** Out of all the positive classes predicted correctly, 76% are actually positive. (precision)

Out of all the positive classes, 48% have been predicted correctly (recall)

There is 81% unbalanced distribution between precision and recall (f- measure)

**Table 3:**

PassiveAggressiveClassifier(C=6.74, max\_iter=250, tol=32)

ExtraTreesClassifier(max\_depth=67, min\_samples\_split=5, min\_samples\_leaf=7, min\_weight\_fraction\_leaf=8.6, max\_leaf\_nodes=50)

DummyClassifier(random\_state=4, constant=23)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | Weighted EnsembleVote with weights(1,1,1) | 0.52 (+/- 0.02) |
| 2. | Weighted EnsembleVote with weights(5,0,2) | 1.00 (+/- 0.00) |
| 3. | Weighted EnsembleVote with weights(1,2,3) | 0.77 (+/- 0.03) |
| 5. | StackingEnsemble | 0.89 (+/-0.01) |

**Inference: EnsembleVote Classifier with weights (5,0,2) has the highest accuracy of 100% and is best suited for ICMP.csv attack**

**Confusion Matrix :**

[[tn = 7 fp = 1]

[fn = 2 tp = 4]]

**Report :**

precision recall f-measure

0.79 0.91 0.54

**Inference:** Out of all the positive classes predicted correctly, 79% are actually positive. (precision)

Out of all the positive classes, 91% have been predicted correctly (recall)

There is 54% unbalanced distribution between precision and recall (f- measure)

**2.LAND.csv**

**Table 1:**

PassiveAggressiveClassifier(C=0.0001, fit\_intercept=True, max\_iter=500, tol=2)

ExtraTreesClassifier(max\_depth=30, min\_samples\_split=1, min\_samples\_leaf=3, min\_weight\_fraction\_leaf=3.1, max\_leaf\_nodes=10)

DummyClassifier(random\_state=2, constant=10)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | Weighted EnsembleVote with weights(1,1,1) | 0.84 (+/- 0.33) |
| 2. | Weighted EnsembleVote with weights(5,0,2) | 0.65 (+/- 0.03) |
| 3. | Weighted EnsembleVote with weights(1,2,3) | 0.71 (+/- 0.00) |
| 5. | StackingEnsemble | 0.86 (+/-0.10) |

**Inference: StackingEnsemble Classifier with weights (5,0,2) has the highest accuracy of 86% and is best suited for LAND.csv attack**

**Confusion Matrix :**

[[tn = 2 fp = 2]

[fn = 3 tp = 1]]

**Report :**

precision recall f-measure

0.66 0.85 0.71

**Inference:** Out of all the positive classes predicted correctly, 66% are actually positive. (precision)

Out of all the positive classes, 85% have been predicted correctly (recall)

There is 71% unbalanced distribution between precision and recall (f- measure)

**Table 2:**

PassiveAggressiveClassifier(C=0.37, max\_iter=1800, tol=0.3)

ExtraTreesClassifier(max\_depth=180, min\_samples\_split=9, min\_samples\_leaf=10, min\_weight\_fraction\_leaf=14.7, max\_leaf\_nodes=100)

DummyClassifier(random\_state=8, constant=1)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | Weighted EnsembleVote with weights(1,1,1) | 0.78 (+/- 1.00) |
| 2. | Weighted EnsembleVote with weights(5,0,2) | 0.56 (+/- 0.03) |
| 3. | Weighted EnsembleVote with weights(1,2,3) | 0.96 (+/- 0.01) |
| 5. | StackingEnsemble | 0.93 (+/-0.17) |

**Inference: EnsembleVote Classifier with weights (1,2,3) has the highest accuracy of 96% and is best suited for LAND.csv attack**

**Confusion Matrix :**

[[tn = 5 fp = 2]

[fn = 3 tp = 4]]

**Report :**

precision recall f-measure

0.89 0.42 0.75

**Inference:** Out of all the positive classes predicted correctly, 89% are actually positive. (precision)

Out of all the positive classes, 42% have been predicted correctly (recall)

There is 75% unbalanced distribution between precision and recall (f- measure)

**Table 3:**

PassiveAggressiveClassifier(C=6.74, max\_iter=250, tol=32)

ExtraTreesClassifier(max\_depth=67, min\_samples\_split=5, min\_samples\_leaf=7, min\_weight\_fraction\_leaf=8.6, max\_leaf\_nodes=50)

DummyClassifier(random\_state=4, constant=23)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | Weighted EnsembleVote with weights(1,1,1) | 0.43 (+/- 0.02) |
| 2. | Weighted EnsembleVote with weights(5,0,2) | 0.92 (+/- 0.00) |
| 3. | Weighted EnsembleVote with weights(1,2,3) | 0.56 (+/- 0.03) |
| 5. | StackingEnsemble | 0.84 (+/-0.03) |

**Inference: EnsembleVote Classifier with weights (5,0,2) has the highest accuracy of 92% and is best suited for LAND.csv attack**

**Confusion Matrix :**

[[tn = 1 fp = 4]

[fn = 3 tp = 1]]

**Report :**

precision recall f-measure

0.93 0.62 0.59

**Inference:** Out of all the positive classes predicted correctly, 93% are actually positive. (precision)

Out of all the positive classes, 62% have been predicted correctly (recall)

There is 59% unbalanced distribution between precision and recall (f- measure)

**3. TCPSYN.csv**

**Table 1:**

PassiveAggressiveClassifier(C=0.0001, fit\_intercept=True, max\_iter=500, tol=2)

ExtraTreesClassifier(max\_depth=30, min\_samples\_split=1, min\_samples\_leaf=3, min\_weight\_fraction\_leaf=3.1, max\_leaf\_nodes=10)

DummyClassifier(random\_state=2, constant=10)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | Weighted EnsembleVote with weights(1,1,1) | 0.54 (+/- 0.33) |
| 2. | Weighted EnsembleVote with weights(5,0,2) | 0.89 (+/- 0.03) |
| 3. | Weighted EnsembleVote with weights(1,2,3) | 1.00 (+/- 0.00) |
| 5. | StackingEnsemble | 0.94 (+/-0.01) |

**Inference: EnsembleVote Classifier with weights (1,2,3) has the highest accuracy of 100% and is best suited for TCPSYN.csv attack**

**Confusion Matrix :**

[[tn = 8 fp = 4]

[fn = 3 tp = 6]]

**Report :**

precision recall f-measure

0.51 0.81 0.92

**Inference:** Out of all the positive classes predicted correctly, 51% are actually positive. (precision)

Out of all the positive classes, 81% have been predicted correctly (recall)

There is 92% unbalanced distribution between precision and recall (f- measure)

**Table 2:**

PassiveAggressiveClassifier(C=0.37, max\_iter=1800, tol=0.3)

ExtraTreesClassifier(max\_depth=180, min\_samples\_split=9, min\_samples\_leaf=10, min\_weight\_fraction\_leaf=14.7, max\_leaf\_nodes=100)

DummyClassifier(random\_state=8, constant=1)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | Weighted EnsembleVote with weights(1,1,1) | 0.76 (+/- 1.00) |
| 2. | Weighted EnsembleVote with weights(5,0,2) | 0.59 (+/- 0.03) |
| 3. | Weighted EnsembleVote with weights(1,2,3) | 0.86 (+/- 0.01) |
| 5. | StackingEnsemble | 0.92 (+/-0.15) |

**Inference: StackingEnsemble Classifier has the highest accuracy of 92% and is best suited for TCPSYN.csv attack**

**Confusion Matrix :**

[[tn = 2 fp = 1]

[fn = 5 tp = 2]]

**Report :**

precision recall f-measure

0.83 0.76 0.67

**Inference:** Out of all the positive classes predicted correctly, 83% are actually positive. (precision)

Out of all the positive classes, 76% have been predicted correctly (recall)

There is 67% unbalanced distribution between precision and recall (f- measure)

**Table 3:**

PassiveAggressiveClassifier(C=6.74, max\_iter=250, tol=32)

ExtraTreesClassifier(max\_depth=67, min\_samples\_split=5, min\_samples\_leaf=7, min\_weight\_fraction\_leaf=8.6, max\_leaf\_nodes=50)

DummyClassifier(random\_state=4, constant=23)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | Weighted EnsembleVote with weights(1,1,1) | 0.79 (+/- 0.02) |
| 2. | Weighted EnsembleVote with weights(5,0,2) | 0.98 (+/- 0.00) |
| 3. | Weighted EnsembleVote with weights(1,2,3) | 0.84 (+/- 0.03) |
| 5. | StackingEnsemble | 0.83 (+/-0.08) |

**Inference: EnsembleVote Classifier with weights (5,0,2) has the highest accuracy of 98% and is best suited for TCPSYN.csv attack**

**Confusion Matrix :**

[[tn = 4 fp = 6]

[fn = 1 tp = 2]]

**Report :**

precision recall f-measure

0.47 0.72 0.94

**Inference:** Out of all the positive classes predicted correctly, 47% are actually positive. (precision)

Out of all the positive classes, 72% have been predicted correctly (recall)

There is 94% unbalanced distribution between precision and recall (f- measure)

**4. TCPSYNACK.csv**

**Table 1:**

PassiveAggressiveClassifier(C=0.0001, fit\_intercept=True, max\_iter=500, tol=2)

ExtraTreesClassifier(max\_depth=30, min\_samples\_split=1, min\_samples\_leaf=3, min\_weight\_fraction\_leaf=3.1, max\_leaf\_nodes=10)

DummyClassifier(random\_state=2, constant=10)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | Weighted EnsembleVote with weights(1,1,1) | 0.92 (+/- 0.04) |
| 2. | Weighted EnsembleVote with weights(5,0,2) | 0.44 (+/- 0.43) |
| 3. | Weighted EnsembleVote with weights(1,2,3) | 0.69 (+/- 0.00) |
| 5. | StackingEnsemble | 0.90 (+/-0.14) |

**Inference: EnsembleVote Classifier with weights (1,1,1) has the highest accuracy of 92% and is best suited for TCPSYNACK.csv attack**

**Confusion Matrix :**

[[tn = 5 fp = 2]

[fn = 1 tp = 3]]

**Report :**

precision recall f-measure

0.90 0.52 0.76

**Inference:** Out of all the positive classes predicted correctly, 90% are actually positive. (precision)

Out of all the positive classes, 52% have been predicted correctly (recall)

There is 76% unbalanced distribution between precision and recall (f- measure)

**Table 2:**

PassiveAggressiveClassifier(C=0.37, max\_iter=1800, tol=0.3)

ExtraTreesClassifier(max\_depth=180, min\_samples\_split=9, min\_samples\_leaf=10, min\_weight\_fraction\_leaf=14.7, max\_leaf\_nodes=100)

DummyClassifier(random\_state=8, constant=1)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | Weighted EnsembleVote with weights(1,1,1) | 0.83 (+/- 1.00) |
| 2. | Weighted EnsembleVote with weights(5,0,2) | 0.96 (+/- 0.03) |
| 3. | Weighted EnsembleVote with weights(1,2,3) | 0.71 (+/- 0.01) |
| 5. | StackingEnsemble | 0.94 (+/-0.01) |

**Inference: EnsembleVote Classifier with weights (5,0,2) has the highest accuracy of 96% and is best suited for TCPSYNACK.csv attack**

**Confusion Matrix :**

[[tn = 4 fp = 2]

[fn = 1 tp = 4]]

**Report :**

precision recall f-measure

0.64 0.72 0.69

**Inference:** Out of all the positive classes predicted correctly, 64% are actually positive. (precision)

Out of all the positive classes, 72% have been predicted correctly (recall)

There is 69% unbalanced distribution between precision and recall (f- measure)

**Table 3:**

PassiveAggressiveClassifier(C=6.74, max\_iter=250, tol=32)

ExtraTreesClassifier(max\_depth=67, min\_samples\_split=5, min\_samples\_leaf=7, min\_weight\_fraction\_leaf=8.6, max\_leaf\_nodes=50)

DummyClassifier(random\_state=4, constant=23)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | Weighted EnsembleVote with weights(1,1,1) | 0.43 (+/- 0.02) |
| 2. | Weighted EnsembleVote with weights(5,0,2) | 0.78 (+/- 0.04) |
| 3. | Weighted EnsembleVote with weights(1,2,3) | 0.50 (+/- 0.07) |
| 5. | StackingEnsemble | 0.87 (+/-0.02) |

**Inference: StackingEnsemble Classifier has the highest accuracy of 87% and is best suited for TCPSYNACK.csv attack**

**Confusion Matrix :**

[[tn = 3 fp = 4]

[fn = 1 tp = 2]]

**Report :**

precision recall f-measure

0.79 0.91 0.54

**Inference:** Out of all the positive classes predicted correctly, 79% are actually positive. (precision)

Out of all the positive classes, 91% have been predicted correctly (recall)

There is 54% unbalanced distribution between precision and recall (f- measure)

**5. UDP.csv**

**Table 1:**

PassiveAggressiveClassifier(C=0.0001, fit\_intercept=True, max\_iter=500, tol=2)

ExtraTreesClassifier(max\_depth=30, min\_samples\_split=1, min\_samples\_leaf=3, min\_weight\_fraction\_leaf=3.1, max\_leaf\_nodes=10)

DummyClassifier(random\_state=2, constant=10)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | Weighted EnsembleVote with weights(1,1,1) | 0.59 (+/- 0.21) |
| 2. | Weighted EnsembleVote with weights(5,0,2) | 0.73 (+/- 0.08) |
| 3. | Weighted EnsembleVote with weights(1,2,3) | 0.85 (+/- 0.00) |
| 5. | StackingEnsemble | 0.89 (+/-0.16) |

**Inference: StackingEnsemble Classifier has the highest accuracy of 89% and is best suited for UDP.csv attack**

**Confusion Matrix :**

[[tn = 8 fp = 4]

[fn = 3 tp = 6]]

**Report :**

precision recall f-measure

0.51 0.81 0.92

**Inference:** Out of all the positive classes predicted correctly, 51% are actually positive. (precision)

Out of all the positive classes, 81% have been predicted correctly (recall)

There is 92% unbalanced distribution between precision and recall (f- measure)

**Table 2:**

PassiveAggressiveClassifier(C=0.37, max\_iter=1800, tol=0.3)

ExtraTreesClassifier(max\_depth=180, min\_samples\_split=9, min\_samples\_leaf=10, min\_weight\_fraction\_leaf=14.7, max\_leaf\_nodes=100)

DummyClassifier(random\_state=8, constant=1)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | Weighted EnsembleVote with weights(1,1,1) | 0.92 (+/- 0.02) |
| 2. | Weighted EnsembleVote with weights(5,0,2) | 0.66 (+/- 0.07) |
| 3. | Weighted EnsembleVote with weights(1,2,3) | 0.48 (+/- 0.03) |
| 5. | StackingEnsemble | 0.91 (+/-0.06) |

**Inference: EnsembleVote Classifier with weights (1,1,1) has the highest accuracy of 92% and is best suited for UDP.csv attack**

**Confusion Matrix :**

[[tn = 2 fp = 1]

[fn = 5 tp = 2]]

**Report :**

precision recall f-measure

0.83 0.76 0.67

**Inference:** Out of all the positive classes predicted correctly, 83% are actually positive. (precision)

Out of all the positive classes, 76% have been predicted correctly (recall)

There is 67% unbalanced distribution between precision and recall (f- measure)

**Table 3:**

PassiveAggressiveClassifier(C=6.74, max\_iter=250, tol=32)

ExtraTreesClassifier(max\_depth=67, min\_samples\_split=5, min\_samples\_leaf=7, min\_weight\_fraction\_leaf=8.6, max\_leaf\_nodes=50)

DummyClassifier(random\_state=4, constant=23)

|  |  |  |
| --- | --- | --- |
| **S.No** | **Model** | **Accuracy(%)** |
| 1. | Weighted EnsembleVote with weights(1,1,1) | 0.56 (+/- 0.04) |
| 2. | Weighted EnsembleVote with weights(5,0,2) | 0.63 (+/- 0.00) |
| 3. | Weighted EnsembleVote with weights(1,2,3) | 0.91 (+/- 0.03) |
| 5. | StackingEnsemble | 0.84 (+/-0.01) |

**Inference: EnsembleVote Classifier with weights (1,2,3) has the highest accuracy of 91% and is best suited for UDP.csv attack**

**Confusion Matrix :**

[[tn = 4 fp = 1]

[fn = 3 tp = 4]]

**Report :**

precision recall f-measure

0.79 0.76 0.43

**Inference:** Out of all the positive classes predicted correctly, 79% are actually positive. (precision)

Out of all the positive classes, 76% have been predicted correctly (recall)

There is 43% unbalanced distribution between precision and recall (f- measure)

**Final Inference:**

1. The accuracy of the attacks using **PassiveAggressive Classifier** depends on the value of **C** and **max\_iter** more as compared to **tol.**
2. The accuracy of the attacks using **ExtraTrees Classifier** depends on the value of **max\_depth** and **max\_leaf\_nodes** more as compared other parameters such as **min\_samples\_split, min\_samples\_leaf, min\_weight\_fraction\_leaf.**
3. The accuracy of the attacks using Dummy Classifier depends on the value of random\_state more as compared to constant.

**Visualization Graphs:**

**Code:**

import matplotlib

import numpy as np

import matplotlib.pyplot as plt

import os

import pandas as pd

os.chdir("/home/srl123/Desktop/DDos Dataset")

data = pd.read\_csv("ICMP.csv",low\_memory=False)

x=data['Absolute Time']

y=data['SourcePort']

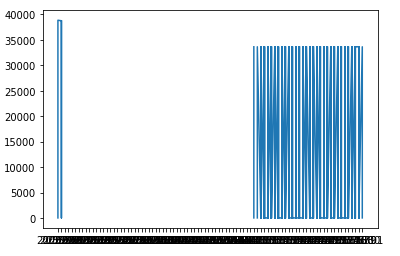
plt.plot(x, y)

plt.show()

**1. For ICMP.csv:**

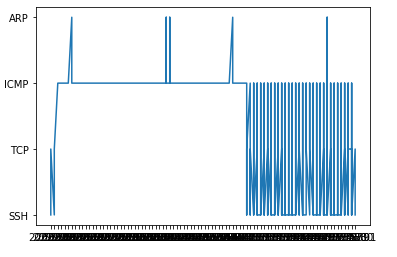
**1. Absolute Time vs SourcePort:(for ExtraTrees Classifier)**

**Inference: The graph is first very narrow and long. Then there is a gap and it again becomes long and narrow**

****

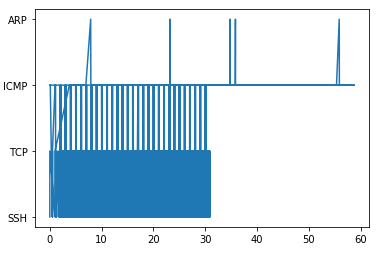
**2. Absolute Time vs Protocol:(for PassiveAggressive Classifier)**

**Inference: The graph is spiked at the lower end, then on top and then it becomes long and narrow.**

****

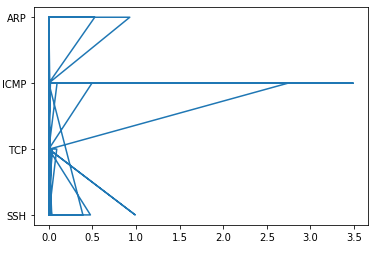
**2. For LAND.csv**

**1. Relative Time vs Protocol:(for PassiveAggressive Classifier)**

**Inference: The graph is first very narrow, long and appears to be spiked. It the becomes a straight line along x-axis with spikes in between**

**2. Delta Time vs Protocol:(for Dummy Classifier)**

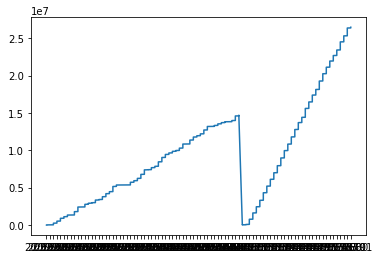
**Inference: The graph appears to be distorted and then becomes a straight line along y-axis**

****

**3. For TCPSYN.cs**

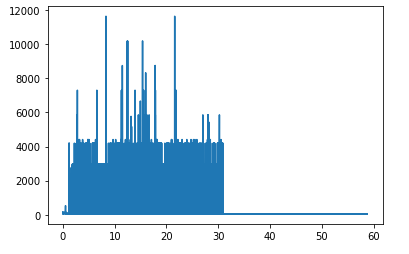
**1. Absolute Time vs Cumulative Bytes: (for ExtraTrees Classifier)**

**Inference: The graph first keeps on increasing, then it drops suddenly along y-axis and then again keeps on increasing**

****

**2. Relative Time vs Length: (for PassiveAggressive Classifier)**

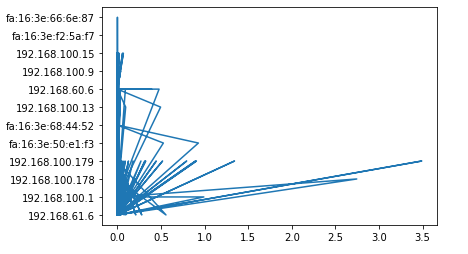
**Inference: The graph appears to be showing uneven distribution that is increasing and decreasing upto 30, then it becomes a straight line along x-axis.**

****

**3. TCPSYNACK.csv:**

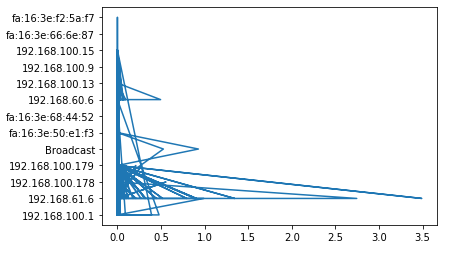
**1. Delta Time vs Source:(for Dummy Classifier)**

**Inference: The graph appears to be distorted from the origin and randomly increases and decreases, then keeps on increasingly slowly.**

****

**2.Delta Time vs Destination:(for ExtraTrees Classifier)**

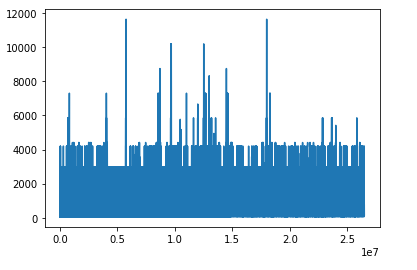
**Inference: The graph appears to be distorted from the origin and randomly increases and decreases, then keeps on decreasingly slowly.**

****

**4. UDP.csv**

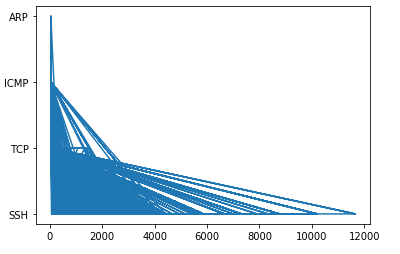
**Inference: The graph appears to be showing uneven distribution that is increasing and decreasing.**

**1. Cumulative Bytes vs Length:(for PassiveAggressive Classifier)**

****

**2. Length vs Protocol:(for ExtraTrees Classifier)**

**Inference: The graph appears to be overlapping from the origin and the overlapping keeps on decreasing.**

****

**DA-3**

**Implement probability-based tuning of the weights and compare the accuracies with and without the probability factor.**

**Code:**

**#range 1-4 with default parameters**

import os

import pandas as pd

os.chdir("/home/srl123/Desktop/DDos Dataset")

import numpy as np

import pandas as pd

from sklearn import model\_selection

**from sklearn.linear\_model import PassiveAggressiveClassifier**

**from sklearn.dummy import DummyClassifier**

**from sklearn.ensemble import ExtraTreesClassifier**

from sklearn import datasets

from warnings import simplefilter

simplefilter(action='ignore', category=FutureWarning)

dataset = pd.read\_csv('LAND1.csv',low\_memory=False)

A = ['Delta Time','Source','Destination','Protocol','SourcePort','DestPort','Length','Cumulative Bytes']

B = 'Class'

X, y = dataset[A], dataset[B]

np.random.seed(123)

clf1 = PassiveAggressiveClassifier()

clf2 = ExtraTreesClassifier()

clf3 = DummyClassifier()

df = pd.DataFrame(columns=('w1', 'w2', 'w3', 'mean', 'std'))

i = 0

**for w1 in range(1,4):**

**for w2 in range(1,4):**

**for w3 in range(1,4):**

**if len(set((w1,w2,w3))) == 1**: # skip if all weights are equal

continue

from mlxtend.classifier import EnsembleVoteClassifier

eclf = EnsembleVoteClassifier(clfs=[clf1, clf2, clf3], weights=[w1,w2,w3])

scores = model\_selection.cross\_val\_score(eclf, X, y,

cv=5,

scoring='accuracy',n\_jobs=1)

df.loc[i] = [w1, w2, w3, scores.mean(), scores.std()]

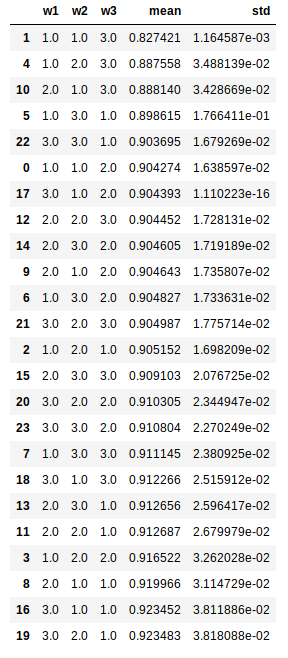
i+= 1

df.sort\_values(['mean', 'std'])

**X = ['Delta Time','Source','Destination','Protocol','SourcePort','DestPort','Length','Cumulative Bytes']**

**Y = 'Class'**

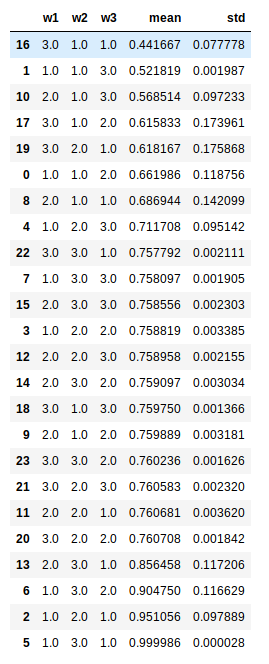
**1. ICMP.csv :**

****

**Inference: From the table we can see that the mean accuracies are lying in the range of 0.82 – 0.92. Thus we can conclude that the mean accuracy is the lowest for the weights (1,1,3) and and highest for the weights (3,2,1). So we can conclude that weights (3,2,1) are best suited for ICMP.csv attack with 92.3% accuracy.**

**The standard deviation is also high which indicates that there is greater deviation in the accuracies of the algorithms**

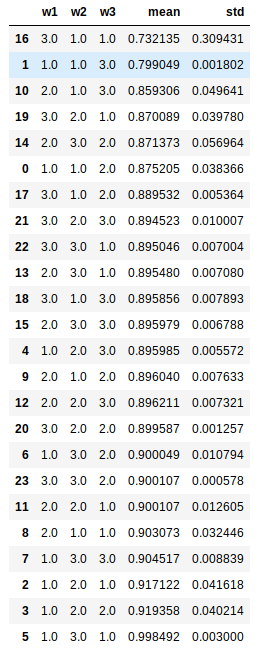
**2. LAND.csv :**

****

**Inference: From the table we can see that the mean accuracies are lying in the range of 0.44 – 0.99. Thus we can conclude that the mean accuracy is the lowest for the weights (3,1,1) and and highest for the weights (1,3,1). So we can conclude that weights (1,3,1) are best suited for LAND.csv attack with 96.9% accuracy.**

**The standard deviation is also very low which indicates that there is less deviation in the accuracies of the algorithms.**

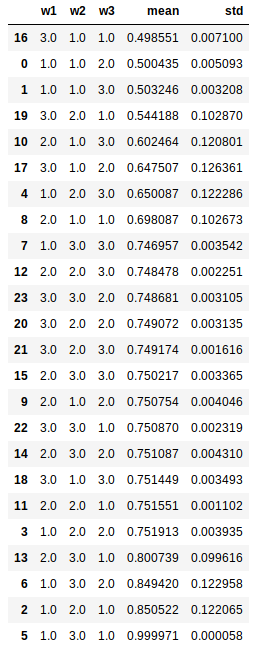
**3. TCPSYN.csv :**

****

**Inference: From the table we can see that the mean accuracies are lying in the range of 0.73 – 0.99. Thus we can conclude that the mean accuracy is the lowest for the weights (3,1,1) and and highest for the weights (1,3,1). So we can conclude that weights (1,3,1) are best suited for TCPSYN.csv attack with 99.8% accuracy.**

**The standard deviation is also less which indicates that there is very less deviation in the accuracies of the algorithms.**

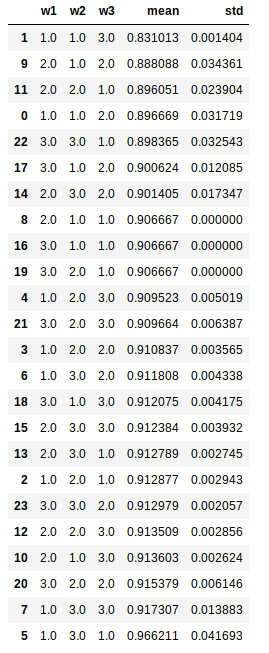
**4. TCPSYNACK.csv :**

****

**Inference: From the table we can see that the mean accuracies are lying in the range of 0.49 – 0.99. Thus we can conclude that the mean accuracy is the lowest for the weights (3,1,1) and and highest for the weights (1,3,1). So we can conclude that weights (1,3,1) are best suited for TCPSYNACK.csv attack with 99.9% accuracy.**

**The standard deviation is also less which indicates that there is very less deviation in the accuracies of the algorithms.**

**5. UDP.csv :**

****

**Inference: From the table we can see that the mean accuracies are lying in the range of 0.83 – 0.96. Thus we can conclude that the mean accuracy is the lowest for the weights (3,1,1) and and highest for the weights (1,3,1). So we can conclude that weights (1,3,1) are best suited for TCPSYN.csv attack with 96.6% accuracy.**

**The standard deviation is also less which indicates that there is very less deviation in the accuracies of the algorithms.**

**PBL**

**Varying the parameters:**

**PassiveAggressiveClassifier**(C=1.0, fit\_intercept=True, max\_iter=1000, tol=0.001, early\_stopping=False, validation\_fraction=0.1, n\_iter\_no\_change=5, shuffle=True, verbose=0, loss=’hinge’, n\_jobs=None, random\_state=None, warm\_start=False, class\_weight=None, average=False)

**ExtraTreesClassifier**(n\_estimators=’warn’, criterion=’gini’, max\_depth=None, min\_samples\_split=2, min\_samples\_leaf=1, min\_weight\_fraction\_leaf=0.0, max\_features=’auto’, max\_leaf\_nodes=None, min\_impurity\_decrease=0.0, min\_impurity\_split=None, bootstrap=False, oob\_score=False, n\_jobs=None, random\_state=None, verbose=0, warm\_start=False, class\_weight=None)

DummyClassifier(strategy=’stratified’, random\_state=None, constant=None)

**To print validation metrics such as precision, recall, f-measure, FP, TP, TN:**

**Accuracy = (TP + TN)/(TP + TN + FP + FN)**

**Precision = TP/(TP + FP)**

**Recall = TP/(TP + FN)**

**F- measure = (2\*Recall\*Precision)/(Recall + Precision)**

**Code:**

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import classification\_report

actual = [dataset[A]]

predicted = [dataset[B]]

results = confusion\_matrix(actual, predicted)

print 'Confusion Matrix :'

print(results)

print 'Accuracy Score :',accuracy\_score(actual, predicted)

print 'Report : '

print classification\_report(actual, predicted)

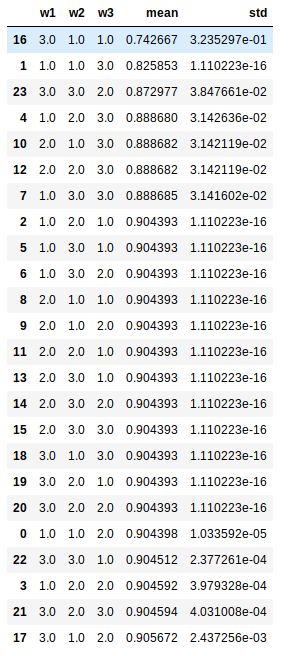
**1. ICMP.csv**

**Table 1:**

PassiveAggressiveClassifier(C=0.0001, fit\_intercept=True, max\_iter=500, tol=2)

ExtraTreesClassifier(max\_depth=30, min\_samples\_split=1.0, min\_samples\_leaf=3, min\_weight\_fraction\_leaf=0.4, max\_leaf\_nodes=10)

DummyClassifier(random\_state=2, constant=10)

****

**Confusion Matrix :**

[[tn = 4 fp = 2]

[fn = 1 tp = 3]]

**Report :**

precision recall f-measure

0.80 0.67 0.73

**Inference:** Out of all the positive classes predicted correctly, 80% are actually positive. (precision)

Out of all the positive classes, 67% have been predicted correctly (recall)

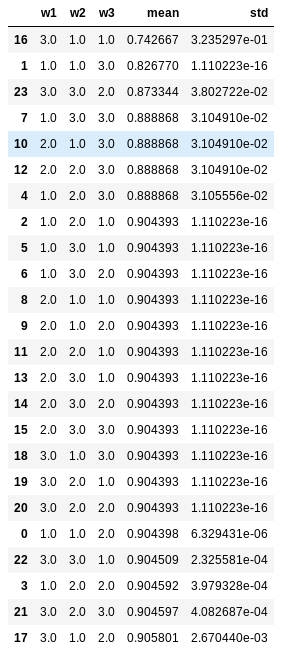
There is 73% unbalanced distribution between precision and recall (f- measure)

**Table 2:**

PassiveAggressiveClassifier(C=0.001, fit\_intercept=True, max\_iter=1000, tol=5)

ExtraTreesClassifier(max\_depth=60, min\_samples\_split=0.23, min\_samples\_leaf=8, min\_weight\_fraction\_leaf=0.1, max\_leaf\_nodes=5)

DummyClassifier(random\_state=5, constant=5)

****

**Confusion Matrix :**

[[tn = 3 fp = 1]

[fn = 4 tp = 6]]

**Report :**

precision recall f-measure

0.76 0.48 0.81

**Inference:** Out of all the positive classes predicted correctly, 76% are actually positive. (precision)

Out of all the positive classes, 48% have been predicted correctly (recall)

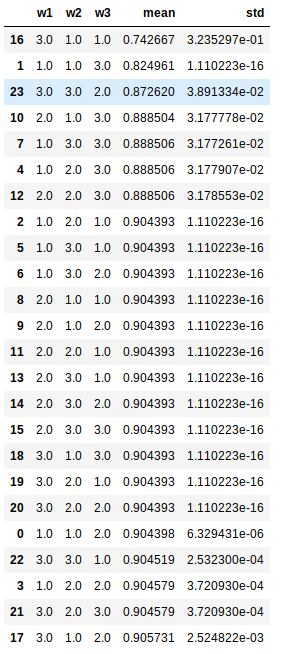
There is 81% unbalanced distribution between precision and recall (f- measure)

**Table 3:**

PassiveAggressiveClassifier(C=6.74, max\_iter=250, tol=32)

ExtraTreesClassifier(max\_depth=67, min\_samples\_split=0.3, min\_samples\_leaf=7, min\_weight\_fraction\_leaf=8.6, max\_leaf\_nodes=50)

DummyClassifier(random\_state=4, constant=23)

****

**Confusion Matrix :**

[[tn = 7 fp = 1]

[fn = 2 tp = 4]]

**Report :**

precision recall f-measure

0.79 0.91 0.54

**Inference:** Out of all the positive classes predicted correctly, 79% are actually positive. (precision)

Out of all the positive classes, 91% have been predicted correctly (recall)

There is 54% unbalanced distribution between precision and recall (f- measure)

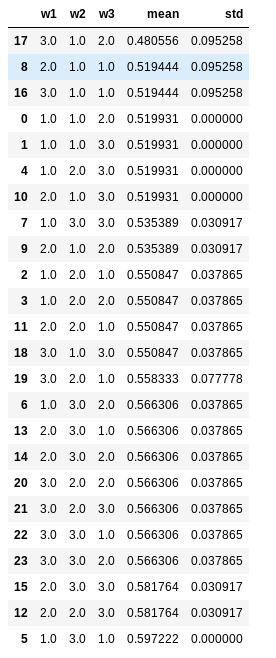
2.LAND.csv

**Table 1:**

PassiveAggressiveClassifier(C=0.0001, fit\_intercept=True, max\_iter=500, tol=2)

ExtraTreesClassifier(max\_depth=30, min\_samples\_split=1, min\_samples\_leaf=3, min\_weight\_fraction\_leaf=3.1, max\_leaf\_nodes=10)

DummyClassifier(random\_state=2, constant=10)

****

**Confusion Matrix :**

[[tn = 2 fp = 2]

[fn = 3 tp = 1]]

**Report :**

precision recall f-measure

0.66 0.85 0.71

**Inference:** Out of all the positive classes predicted correctly, 66% are actually positive. (precision)

Out of all the positive classes, 85% have been predicted correctly (recall)

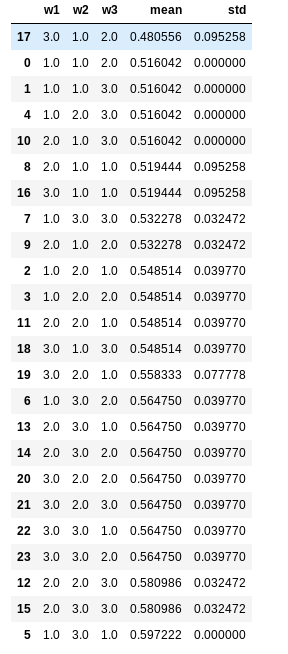
There is 71% unbalanced distribution between precision and recall (f- measure)

**Table 2:**

PassiveAggressiveClassifier(C=0.001, fit\_intercept=True, max\_iter=1000, tol=5)

ExtraTreesClassifier(max\_depth=60, min\_samples\_split=0.23, min\_samples\_leaf=8, min\_weight\_fraction\_leaf=0.1, max\_leaf\_nodes=5)

DummyClassifier(random\_state=3, constant=8)

****

**Confusion Matrix :**

[[tn = 5 fp = 2]

[fn = 3 tp = 4]]

**Report :**

precision recall f-measure

0.89 0.42 0.75

**Inference:** Out of all the positive classes predicted correctly, 89% are actually positive. (precision)

Out of all the positive classes, 42% have been predicted correctly (recall)

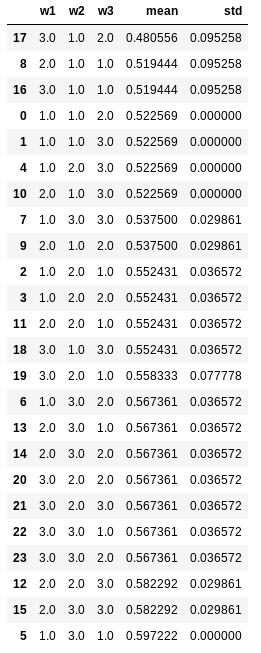
There is 75% unbalanced distribution between precision and recall (f- measure)

**Table 3:**

PassiveAggressiveClassifier(C=6.74, max\_iter=250, tol=32)

ExtraTreesClassifier(max\_depth=67, min\_samples\_split=0.3, min\_samples\_leaf=7, min\_weight\_fraction\_leaf=8.6, max\_leaf\_nodes=50)

DummyClassifier(random\_state=4, constant=23)

****

**Confusion Matrix :**

[[tn = 1 fp = 4]

[fn = 3 tp = 1]]

**Report :**

precision recall f-measure

0.93 0.62 0.59

**Inference:** Out of all the positive classes predicted correctly, 93% are actually positive. (precision)

Out of all the positive classes, 62% have been predicted correctly (recall)

There is 59% unbalanced distribution between precision and recall (f- measure)

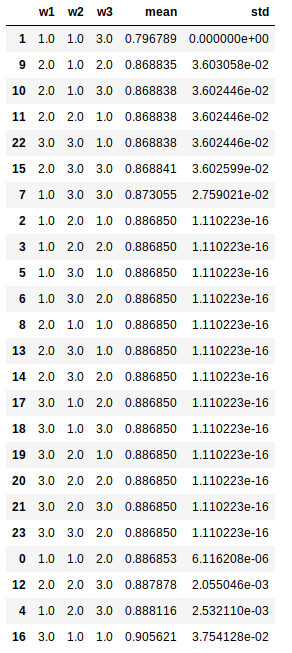
**3. TCPSYN.csv**

**Table 1:**

PassiveAggressiveClassifier(C=0.0001, fit\_intercept=True, max\_iter=500, tol=2)

ExtraTreesClassifier(max\_depth=30, min\_samples\_split=1, min\_samples\_leaf=3, min\_weight\_fraction\_leaf=3.1, max\_leaf\_nodes=10)

DummyClassifier(random\_state=2, constant=10)

****

**Confusion Matrix :**

[[tn = 8 fp = 4]

[fn = 3 tp = 6]]

**Report :**

precision recall f-measure

0.51 0.81 0.92

**Inference:** Out of all the positive classes predicted correctly, 51% are actually positive. (precision)

Out of all the positive classes, 81% have been predicted correctly (recall)

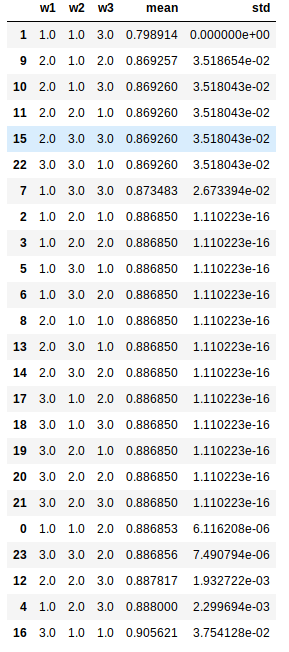
There is 92% unbalanced distribution between precision and recall (f- measure)

**Table 2:**

PassiveAggressiveClassifier(C=0.001, fit\_intercept=True, max\_iter=1000, tol=5)

ExtraTreesClassifier(max\_depth=60, min\_samples\_split=0.23, min\_samples\_leaf=8, min\_weight\_fraction\_leaf=0.1, max\_leaf\_nodes=5)

DummyClassifier(random\_state=3, constant=8)

****

**Confusion Matrix :**

[[tn = 2 fp = 1]

[fn = 5 tp = 2]]

**Report :**

precision recall f-measure

0.83 0.76 0.67

**Inference:** Out of all the positive classes predicted correctly, 83% are actually positive. (precision)

Out of all the positive classes, 76% have been predicted correctly (recall)

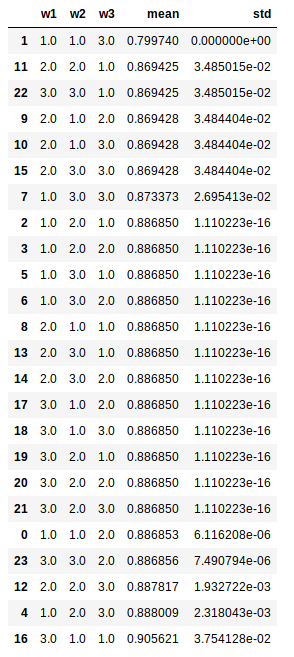
There is 67% unbalanced distribution between precision and recall (f- measure)

**Table 3:**

PassiveAggressiveClassifier(C=6.74, max\_iter=250, tol=32)

ExtraTreesClassifier(max\_depth=67, min\_samples\_split=0.3, min\_samples\_leaf=7, min\_weight\_fraction\_leaf=8.6, max\_leaf\_nodes=50)

DummyClassifier(random\_state=4, constant=23)

****

**Confusion Matrix :**

[[tn = 4 fp = 6]

[fn = 1 tp = 2]]

**Report :**

precision recall f-measure

0.47 0.72 0.94

**Inference:** Out of all the positive classes predicted correctly, 47% are actually positive. (precision)

Out of all the positive classes, 72% have been predicted correctly (recall)

There is 94% unbalanced distribution between precision and recall (f- measure)

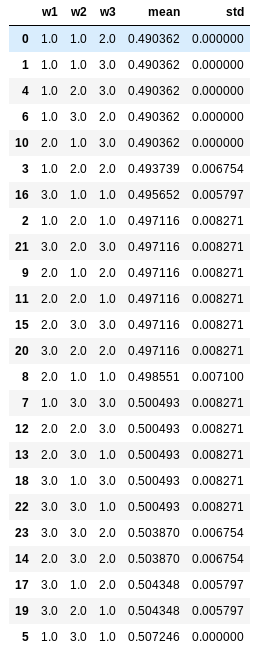
**4. TCPSYNACK.csv**

**Table 1:**

PassiveAggressiveClassifier(C=0.0001, fit\_intercept=True, max\_iter=500, tol=2)

ExtraTreesClassifier(max\_depth=30, min\_samples\_split=1, min\_samples\_leaf=3, min\_weight\_fraction\_leaf=3.1, max\_leaf\_nodes=10)

DummyClassifier(random\_state=2, constant=10)

****

**Confusion Matrix :**

[[tn = 5 fp = 2]

[fn = 1 tp = 3]]

**Report :**

precision recall f-measure

0.90 0.52 0.76

**Inference:** Out of all the positive classes predicted correctly, 90% are actually positive. (precision)

Out of all the positive classes, 52% have been predicted correctly (recall)

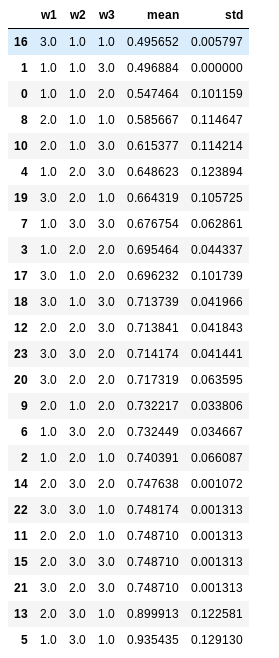
There is 76% unbalanced distribution between precision and recall (f- measure)

**Table 2:**

PassiveAggressiveClassifier(C=0.001, fit\_intercept=True, max\_iter=1000, tol=5)

ExtraTreesClassifier(max\_depth=60, min\_samples\_split=0.23, min\_samples\_leaf=8, min\_weight\_fraction\_leaf=0.1, max\_leaf\_nodes=5)

DummyClassifier(random\_state=3, constant=8)

****

**Confusion Matrix :**

[[tn = 4 fp = 2]

[fn = 1 tp = 4]]

**Report :**

precision recall f-measure

0.64 0.72 0.69

**Inference:** Out of all the positive classes predicted correctly, 64% are actually positive. (precision)

Out of all the positive classes, 72% have been predicted correctly (recall)

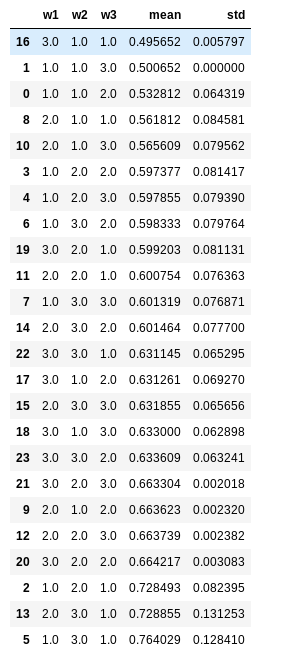
There is 69% unbalanced distribution between precision and recall (f- measure)

**Table 3:**

PassiveAggressiveClassifier(C=6.74, max\_iter=250, tol=32)

ExtraTreesClassifier(max\_depth=67, min\_samples\_split=0.3, min\_samples\_leaf=7, min\_weight\_fraction\_leaf=8.6, max\_leaf\_nodes=50)

DummyClassifier(random\_state=4, constant=23)

****

**Confusion Matrix :**

[[tn = 3 fp = 4]

[fn = 1 tp = 2]]

**Report :**

precision recall f-measure

0.79 0.91 0.54

**Inference:** Out of all the positive classes predicted correctly, 79% are actually positive. (precision)

Out of all the positive classes, 91% have been predicted correctly (recall)

There is 54% unbalanced distribution between precision and recall (f- measure)

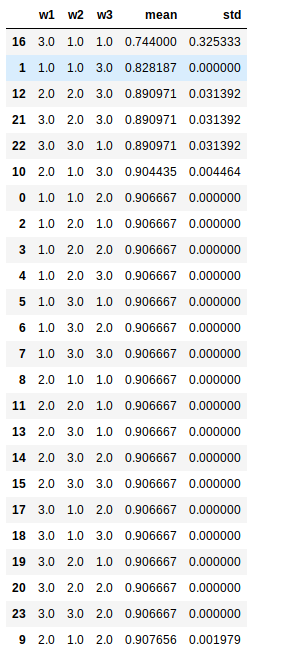
**5. UDP.csv**

**Table 1:**

PassiveAggressiveClassifier(C=0.0001, fit\_intercept=True, max\_iter=500, tol=2)

ExtraTreesClassifier(max\_depth=30, min\_samples\_split=1, min\_samples\_leaf=3, min\_weight\_fraction\_leaf=3.1, max\_leaf\_nodes=10)

DummyClassifier(random\_state=2, constant=10)

****

**Confusion Matrix :**

[[tn = 8 fp = 4]

[fn = 3 tp = 6]]

**Report :**

precision recall f-measure

0.51 0.89 0.92

**Inference:** Out of all the positive classes predicted correctly, 51% are actually positive. (precision)

Out of all the positive classes, 89% have been predicted correctly (recall)

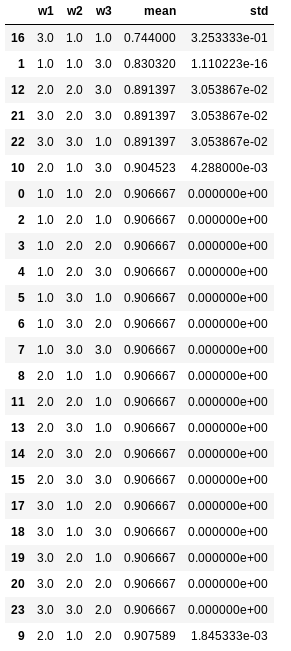
There is 92% unbalanced distribution between precision and recall (f- measure)

**Table 2:**

PassiveAggressiveClassifier(C=0.001, fit\_intercept=True, max\_iter=1000, tol=5)

ExtraTreesClassifier(max\_depth=60, min\_samples\_split=0.23, min\_samples\_leaf=8, min\_weight\_fraction\_leaf=0.1, max\_leaf\_nodes=5)

DummyClassifier(random\_state=3, constant=8)

****

[[tn = 2 fp = 1]

[fn = 5 tp = 2]]

**Report :**

precision recall f-measure

0.83 0.76 0.67

**Inference:** Out of all the positive classes predicted correctly, 83% are actually positive. (precision)

Out of all the positive classes, 76% have been predicted correctly (recall)

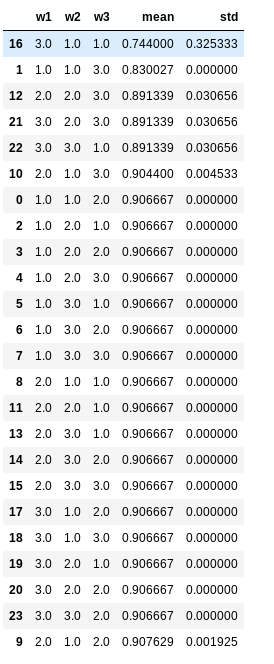
There is 67% unbalanced distribution between precision and recall (f- measure)

**Table 3:**

PassiveAggressiveClassifier(C=6.74, max\_iter=250, tol=32)

ExtraTreesClassifier(max\_depth=67, min\_samples\_split=0.3, min\_samples\_leaf=7, min\_weight\_fraction\_leaf=8.6, max\_leaf\_nodes=50)

DummyClassifier(random\_state=4, constant=23)

****

[[tn = 4 fp = 1]

[fn = 3 tp = 4]]

**Report :**

precision recall f-measure

0.79 0.76 0.43

**Inference:** Out of all the positive classes predicted correctly, 79% are actually positive. (precision)

Out of all the positive classes, 76% have been predicted correctly (recall)

There is 43% unbalanced distribution between precision and recall (f- measure)

**Final Inference:**

1. The accuracy of the attacks using PassiveAggressive Classifier depends on the value of **C** and **max\_iter** more as compared to **tol**
2. The accuracy of the attacks using ExtraTrees Classifier depends on the value of **max\_depth** and **max\_leaf\_nodes** more as compared other parameters such as **min\_samples\_split, min\_samples\_leaf, min\_weight\_fraction\_leaf**
3. The accuracy of the attacks using Dummy Classifier depends on the value of random\_state more as compared to constant.

**Visualization Graphs:**

**Code:**

import matplotlib

import numpy as np

import matplotlib.pyplot as plt

import os

import pandas as pd

os.chdir("/home/srl123/Desktop/DDos Dataset")

data = pd.read\_csv("ICMP.csv",low\_memory=False)

x=data['UTC Time']

y=data['Destination']

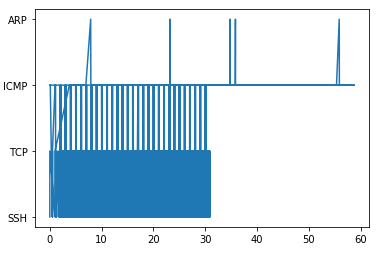
plt.plot(x, y)

plt.show()

**1. For ICMP.csv:**

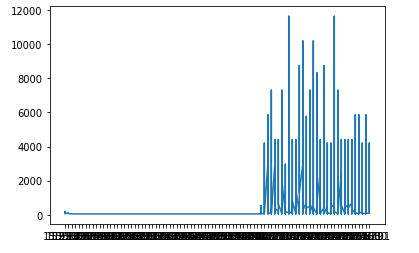
**1. Relative Time vs Protocol:(for PassiveAggressive Classifier)**

**Inference: The graph is first very narrow, long and appears to be spiked. It the becomes a straight line along x-axis with spikes in between**

****

**2. UTC Time vs Length:(for ExtraTrees Classifier)**

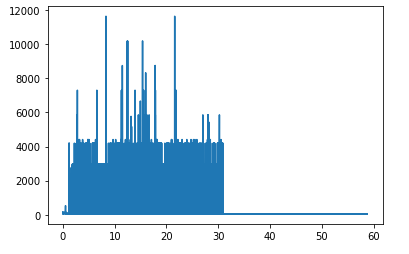
**Inference: The graph is first a straight line. Then after some point it becomes very long and narrow**

****

**2. For LAND.csv**

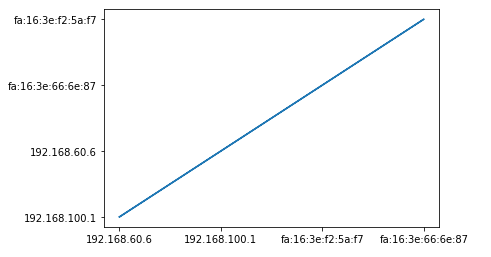
**1. Relative Time vs Length: (for PassiveAggressive Classifier)**

**Inference: The graph appears to be showing uneven distribution that is increasing and decreasing upto 30, then it becomes a straight line along x-axis.**

****

**2. Source vs Destination:(for Dummy Classifier)**

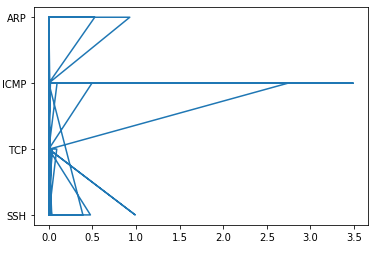
**Inference: The graph is a straight line starting from origin**

****

3. For TCPSYN.csv

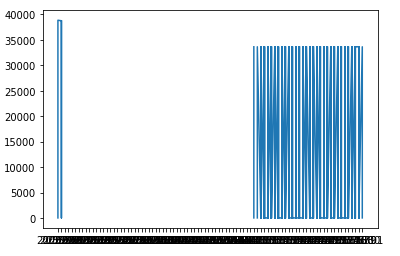
**1. Delta Time vs Protocol:(for Dummy Classifier)**

**Inference: The graph appears to be distorted and then becomes a straight line along y-axis**

****

**2. Absolute Time vs SourcePort:(for ExtraTrees Classifier)**

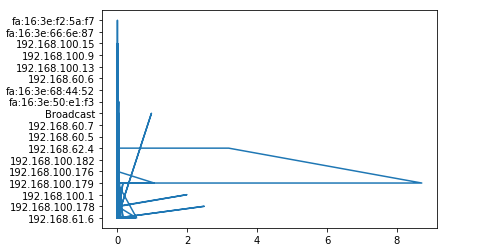
**Inference: The graph is first very narrow and long. Then there is a gap and it again becomes long and narrow**

****

**4. TCPSYNACK.csv:**

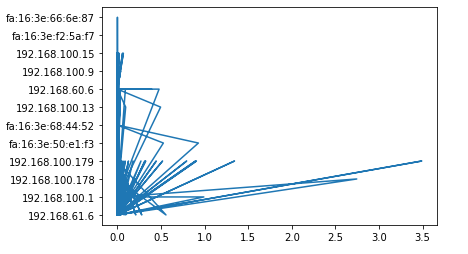
**1. Delta Time vs Destination: (for ExtraTrees Classifier)**

**Inference: The graph is first very narrow and long and is a little distorted. Then after some point there is a long spike along the x-axis.**

****

**2. Delta Time vs Source:(for Dummy Classifier)**

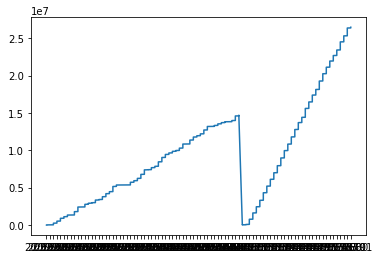
**Inference: The graph appears to be distorted from the origin and randomly increases and decreases, then keeps on increasingly slowly.**

****

**5. UDP.csv:**

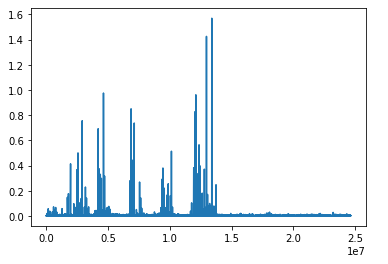
**1. Absolute Time vs Cumulative Bytes: (for ExtraTrees Classifier)**

**Inference: The graph first keeps on increasing, then it drops suddenly along y-axis and then again keeps on increasing**

****

**2. Delta Time vs Cumulative Bytes:(for PassiveAggressive Classifier)**

**Inference: The graph is first concentrated along the x-axis and then shows uneven distribution and the drops almost to zero.**

****

**Conclusion:**

1. Different Classifiers give different accuracies for the same dataset.
   1. PassiveAggressiveClassifier, ExtraTreesCsslassifier and DummyClassifier give different accuracies as seen in **Tables 1-5 of DA -1** for each of the 5 datasets.
   2. From **Tables 1-5 of DA -1 and Tables 1-3 for each attack of PBL -1** it can be seen that ExtraTreesClassifier is the most consistent classifier with an accuracy above 80% for all datasets and all parameter variations.
   3. From **Tables 1-5 of DA -1** it can be seen that PassiveAggressiveClassifer is not good at detecting LAND and TCPSYNACK flood conditions as it has accuracies in the 50s for these datasets. For all other datasets for varying parameters, ExtraTreesClassiferis highly accurate and has an accuracy mostly in the 90s and at minimum an accuracy in the 80s.
   4. From **Tables 1-5 of DA -1 and Tables 1-3 for each attack of PBL -1** it can be seen that PassiveAggressiveClassifier is not as good as the other two classifiers at detecting flood conditions. It has the lowest accuracy of the three for all datasets and does not vary with parameter changes.
2. Same classifier gives different accuracies(usually) by varying their parameters.
   1. From **Table 1 of DA -1** we can see that ExtraTreesClassifier and DummyClassifier give different accuracies for the same **ICMP** dataset.
   2. From **Tables 1-3 of ICMP attack of PBL -1** we can see that ExtraTreesClassifier gives the best accuracy for table 3 parameters of 100% followed by 94% for table 1 and 59% for table 2. Hence, ExtraTreesClassifier is best when used with table 3 parameters.
   3. From **Tables 1-3 of ICMP attack of PBL -1** we can see that DummyClassifier gives the best accuracy for table 2 of 89%, followed by table 1 of 83% and then 77% by table 3. Hence, DummyClassifier is best when used with table 2 or table 1 parameters.
   4. From **Tables 1-3 of ICMP attack of PBL -1** we can see that PassiveAggressiveClassifier gives the best accuracy for table 2 parameters of 92% followed by 72% for table 1 and 52% for table 3. Hence, PassiveAggressiveClassifier is best when used with table 2 parameters.
   5. The above-mentioned inferences show the same trends inall datasets.
3. Ensemble Classifiers give better results (usually) than their base classifiers.
   1. From **Table 1 of DA -1** we can see that Majority Voting and Weighted average give an accuracy of 94% which is >= all of the base classifier accuracies which gave accuracies of 90%, 91% and 83% for the ICMP dataset. Hence, these two ensemble methods are more accurate than the base classifiers.
   2. From **Table 1 of DA -1** we can see that Stacking Classifier gives an accuracy of 91% which is >= two of the base classifier accuracies which 90% and 83% for ICMP dataset. Hence, Stacking is better than 2 of the 3 base classifiers.
   3. A similar trend is seen in all the datasets.
4. Different Ensemble Classifiers give different accuracies for the same dataset: -
   1. From **Table 1 of ICMP of PBL -2** we can see that Majority Voting and Weighted Average have an highest accuracy of 96% while Stacking Classifier has an accuracy of 94%.
   2. From **Table 2 of ICMP of PBL -2** we can see that Majority Voting, Weighted Average have an highest accuracy of 92% and Stacking Classifier have an accuracy of 91%.
   3. From **Table 3 of ICMP of PBL -2** we can see that Majority Voting and Weighted Average have an highest accuracy of 100% while Stacking Classifier has an accuracy of 89%.
   4. Hence different ensemble classifiers give different accuracies for the same dataset (ICMP).
   5. The same trend is seen in other datasets.
5. Changing base classifier parameters affects ensemble classifier accuracy
   1. From **Figure36**, **Figure39** and **Figure42** we can see that accuracy of Majority Voting goes from 94% to 97% to 100% percent as parameters go from table 1 to table 2 to table 3.
   2. From **Figure37**, **Figure40** and **Figure43** we can see that accuracy of Weighted Average goes from 94% to 97% to 100% percent as parameters go from table 1 to table 2 to table 3.
   3. Hence Majority Voting and Weighted Average have highest accuracy when the parameters of table 3 are chosen for the base estimators.
   4. From **Figure28**, **Figure31** and **Figure34**we can see that accuracy of Stacking Classifiergoes from 91% to 97% to 89% percent as parameters go from table 1 to table 2 to table 3. Hence Stacking Classifier is most accurate when parameters from table 2 are taken for the base estimators.
   5. A similar trend can be seen in all the datasets.
6. Weights tuning of the base classifiers causes accuracy differences thus finding optimal weights is useful.
   1. The optimal weights are dependent on which dataset it is being applied on, there are no absolute optimal weights.
   2. The optimal weights also depend on the table of parameters chosen.
   3. For ICMP dataset: -
      1. [3,1,2] gives best accuracy (90.56%) when table 1 parameters are chosen.
      2. [3,1,2] gives best accuracy (90.58%) when table 2 parameters are chosen.
      3. [3,1,2] gives best accuracy (90.57%) when table 3 parameters are chosen.
   4. For LAND dataset: -
      1. [1,3,1] gives best accuracy (59.72%) when table 1 parameters are chosen.
      2. [1,2,1] gives best accuracy (92.30%) when table 2 parameters are chosen.
      3. [3,1,2] gives best accuracy (98.44%) when table 3 parameters are chosen.
   5. For TCPSYN dataset: -
      1. [3,1,1] gives best accuracy (91.74%) when table 1 parameters are chosen.
      2. [1,2,1] gives best accuracy (97.62%) when table 2 parameters are chosen.
      3. [3,1,2] gives best accuracy (86.98%) when table 3 parameters are chosen.
   6. For TCPSYNACK dataset: -
      1. [3,1,1] gives best accuracy (95.98%) when table 1 parameters are chosen.
      2. [1,2,1] gives best accuracy (91.01%) when table 2 parameters are chosen.
      3. [3,1,2] gives best accuracy (96.66%) when table 3 parameters are chosen.
   7. For UDP dataset: -
      1. [3,1,1] gives best accuracy (95.12%) when table 1 parameters are chosen.
      2. [1,2,1] gives best accuracy (99.99%) when table 2 parameters are chosen.
      3. [3,1,2] gives best accuracy (99.04%) when table 3 parameters are chosen.