DFRWS

DIGITAL FORENSIC RESEARCH CONFERENCE

# Machine Learning Based Approach to Analyze File Meta Data for Smart Phone File Triage

By:

Cezar Serhal (University College Dublin) and Nhien-An Le-Khac (University College Dublin)

*From the proceedings of*

The Digital Forensic Research Conference

**DFRWS USA 2021**

July 12-15, 2021

# DFRWS USA 2021

## Paper Presentation

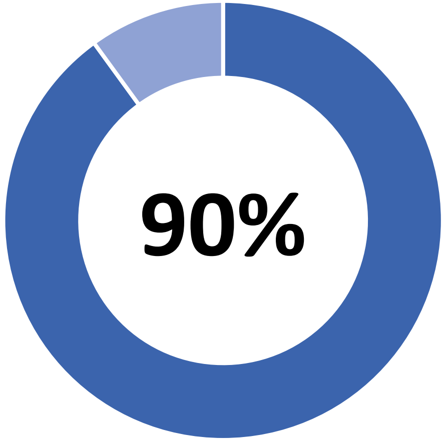
**Paper #32:** Machine Learning Based Approach to Analyze File Metadata for Smartphone File Triage

Cezar Serhal  Nhien-An Le-Khac
University College Dublin

# Content

# Increasing Data Sizes

## Mobile Phone Penetration (2017)




(Deloitte, 2017)

## Cases and Mobile Phones

(Marturana et al., 2011b; Faheem et al. 2014)

## Average Mobile Phone Storage Capacity (2019)

**83 GBs**

(LIM, 2019)

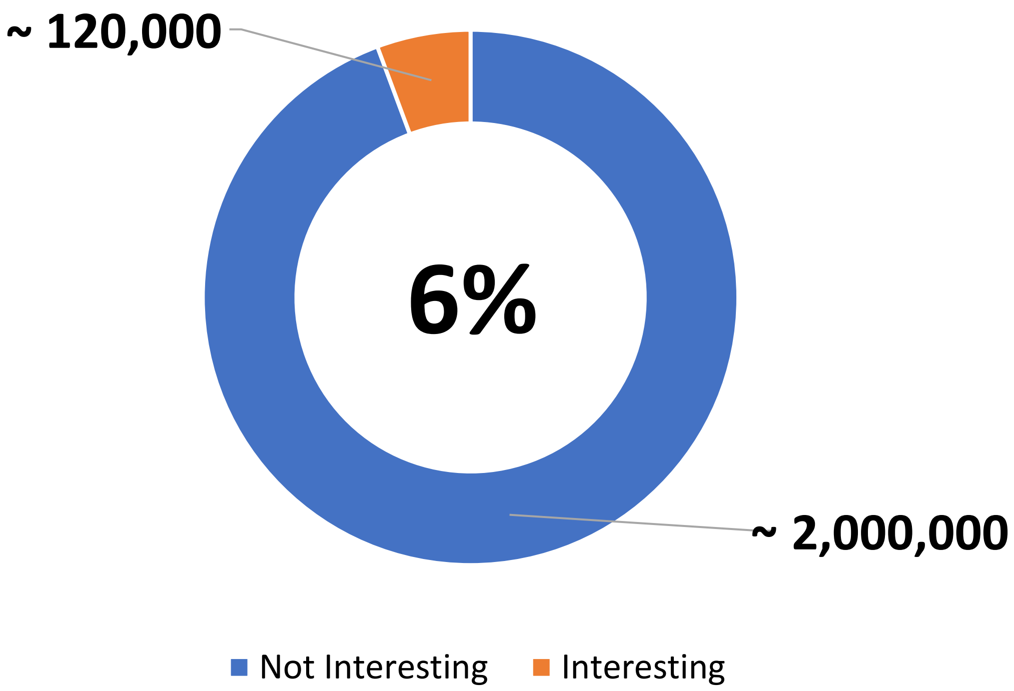
## Focus on Obtaining all Data

(Gómez, 2012; Witteman et al. 2016)

# Resulting Challenges

| | |
|---|---|
| 1 | Difficulty of manually identify relevant examinable files within a plethora of uninteresting OS and application files extracted by forensic tools |
| 2 | Occupying valuable investigation time in examining file of no relevance and build-up of backlog |
| 3 | Ineffectiveness in time critical cases where examiners need to identify evidence in the shortest time possible |

## % Interesting Files in Dataset




Dataset | 12 Phones | 1,998,950 Files

# Triage Techniques to Tackle these Challenges

## Classical Automation Based Approaches

- Automated tools that embed the examiners' knowledge and skills can be used to classify files based on their possible interest.
- Examples of such approaches rely on block hashing and regular expression matching (e.g bulk_extractor presented in (Garfinkel, 2013))
- A significant limitation of such approaches is requiring developers or LE users to know and hardcode data templates and relations of interest.

## ML Based Approaches

- Some researchers classified whether a device is of interest based on its usage metrics and file-system metadata.
- Other researchers applied ML on file and system metadata to identify the owner of carved data or to select relevant events to construct a cybercrime events timeline.
- Only one recent research proposed a ML approach to classify files based on metadata. However, the results are based on data generated and extracted from a computer-based operating system rather than a smart mobile operating system. In addition, the approach can be enhanced by applying feature selection and hyperparameter tuning.

# Content

# Research Objectives

Answering the question of whether file metadata and ML can be used to decide if a file extracted from a smart phone should be examined or not

## Hypothesis

- File metadata can indicate the relevance of a file for an investigation.
- ML classification algorithms can model the decision-making process required to identify files of interest.
- Different classification models will perform differently.

# Content

Problem Statement

Research Objectives

Related Work

Methodology

Results

Conclusion and Future Work

# Machine Learning Approaches for Digital Triage

**(Marturana et al., 2011) – A Quantitative Approach to Triaging in Mobile Forensics [Classify a Device]**

- Device usage information and several classification algorithms are used to determine the likelihood that a device was used to commit a crime related to pedophilia.
- DT performs the best.

**(Gómez, 2012) – Triage in-Lab: case backlog reduction with forensic digital profiling [Classify a Device]**

- Utilized **device usage information** and different ML classification algorithms to identify if an examined device is relevant to a specific crime.
- Experiments on 21 forensic images of hard disk, the best performance is achieved by KNN reporting an accuracy of 90%.

**(Dalins et al., 2018) A labelling schema for child exploitation materials [Classify a File]**

- Designed and tested a deep-learning based child exploitation material classifiers.
- Classifier was sufficient for triaging the existence of child exploitation material, however it did not perform well in classifying the severity of the images against existing scales.

**(Du and Scanlon, 2019) Methodology for the Automated Metadata-Based Classification of Incriminating Digital Forensic Artefacts [Classify a File]**

- Presented a ML based approach for automated identification of incriminating digital forensic artifacts based on **file metadata**.
- Results are based on files generated and extracted from **computers** and not smart mobile phones.
- The adopted approach leaves room for enhancement in some areas such as feature engineering, feature selection, and hyper-parameter tunning.
- Results showed that performance is affected by the prevalence of the class of interest. This highlights the importance of using data similar to real world cases in order to generate classifiers that are effective in practice.

# Applications of ML in Analysing File Metadata in Digital Forensics

**(Khan and Wakeman, 2006) – Machine Learning for Post-Event Timeline Reconstruction**

- Used Recurrent Neural Networks to reconstruct a post-event timeline that in essence identifies which and when applications where run.
- Accuracy of the implemented algorithm increased with increasing the duration of file system activity of the training data. Key limitations of the approach include the need for training a separate neural network for each application or a significantly different version of an

**(Mohammad, 2019) – An Enhanced Multiclass Support Vector Machine Model and its Application to Classifying File Systems Affected by a Digital Crime**

- Used classification algorithms to reconstruct cybercrime events based on file system metadata.
- NN and the RF algorithms achieved the highest precision of 89%.

**(Garfinkel et al., 2011) – An Automated Solution to the Multiuser Carved Data Ascription Problem**

- Presented a solution to identifying the owner of data carved from storage media used by multiple users. Classification algorithms such as KNN and DT are used in conjunction with file system metadata and extended file metadata to calculate the ownership likelihood for each possible user.
- DT performed better than the tried KNN algorithms.

**(Milosevic et al., 2017) – Machine learning aided Android malware classification**

- Implemented two approaches based on ML algorithms including SVM with Sequential Minimal Optimization, NB, DT, JRIP, and logistic regression to detect Android malware.
- Ensemble learning (using AdaBoost) improve performance.

# Content

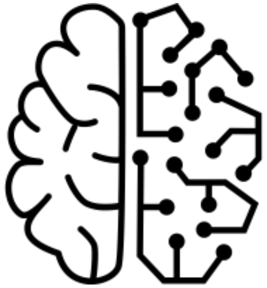
# Scope and Setup

## Toolkit

## Corpus

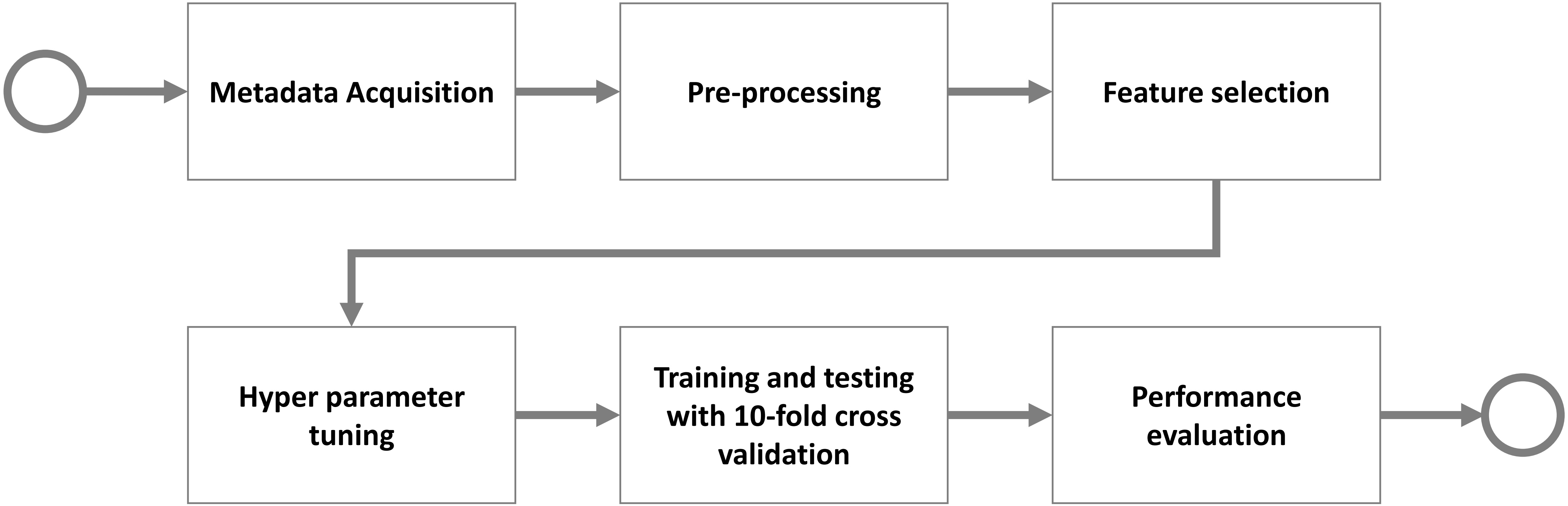
- **12 Android mobile phones**
- **Related to Terrorism Cases**
- **~ 2 Million Files**

## ML Classifiers

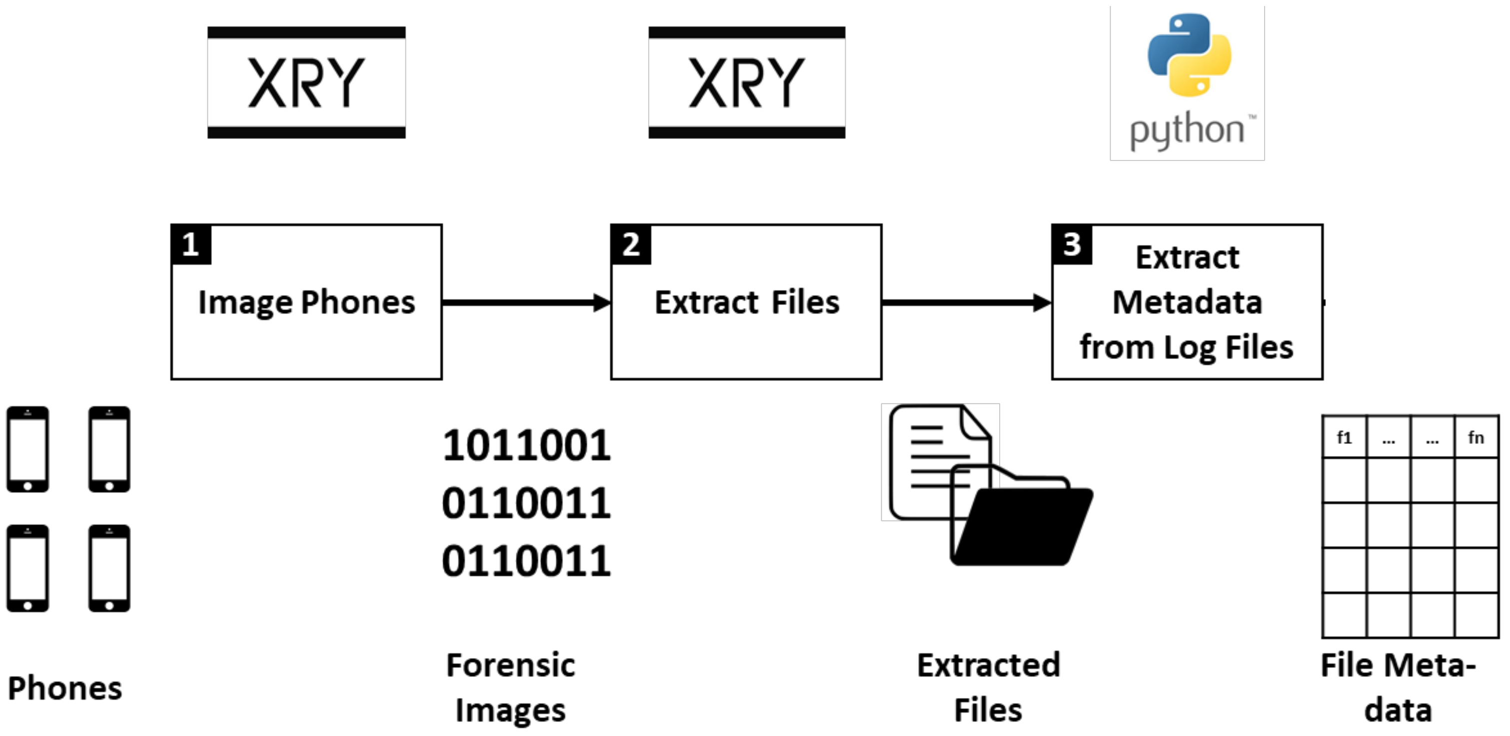


- Naïve Bayes
- K-Nearest Neighbour
- Support Vector Machines
- DT namely Classification and Regression Trees
- Random Forests
- Neural Network – Multi Layer Perceptron

# Approach Overview



# Approach: Metadata Acquisition



# Approach: Pre-processing

1 Explore Data → 2 Create Features → 3 Clean Data → 4 Encode Categorical Features → 5 Normalize Features

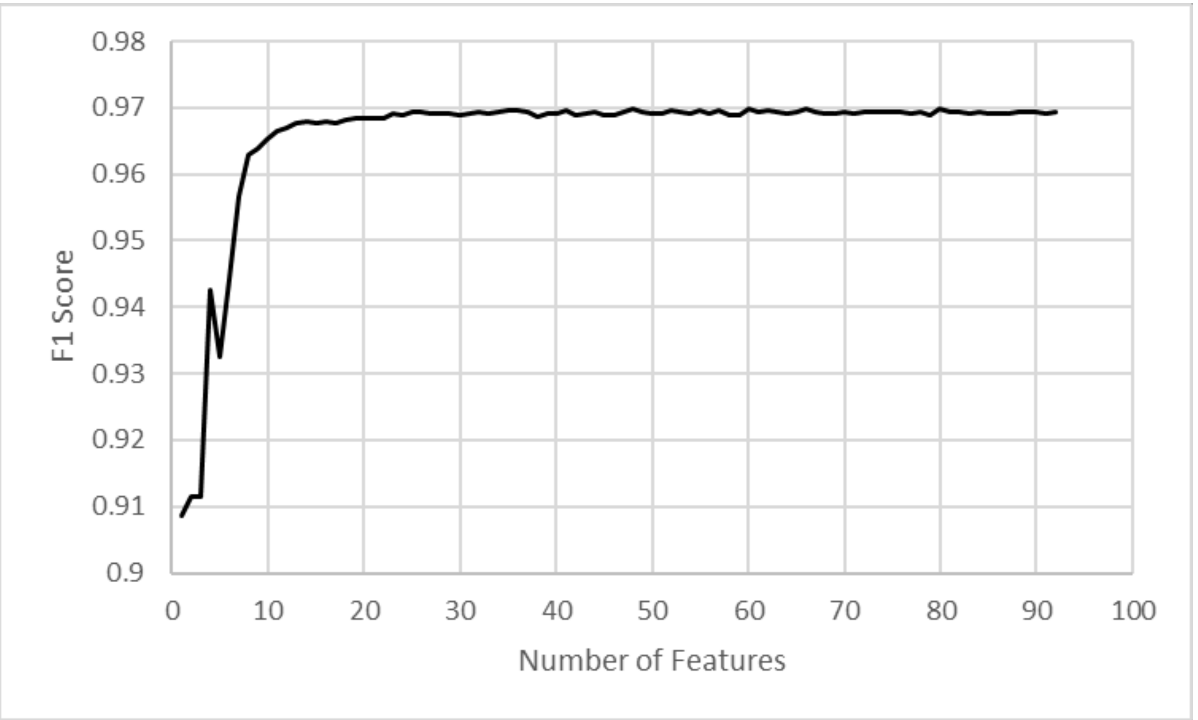
Sample Dependent

Use pipelines to avoid data leakage with cross-validation

# Approach: Feature Selection

**Recursive Feature Elimination and Cross-validation**

**F1-Score vs. Number of Features**




**List of Selected Features**

General:

1. File Type (extension)
2. File Size (KB)
3. Delta Apprehended (Days) (Apprehended date - file modified date)
4. EXIF Flag (existence of EXIF data)
5. Xresolution (Photos only)
6. Yresolution (Photos only)

Filename related:

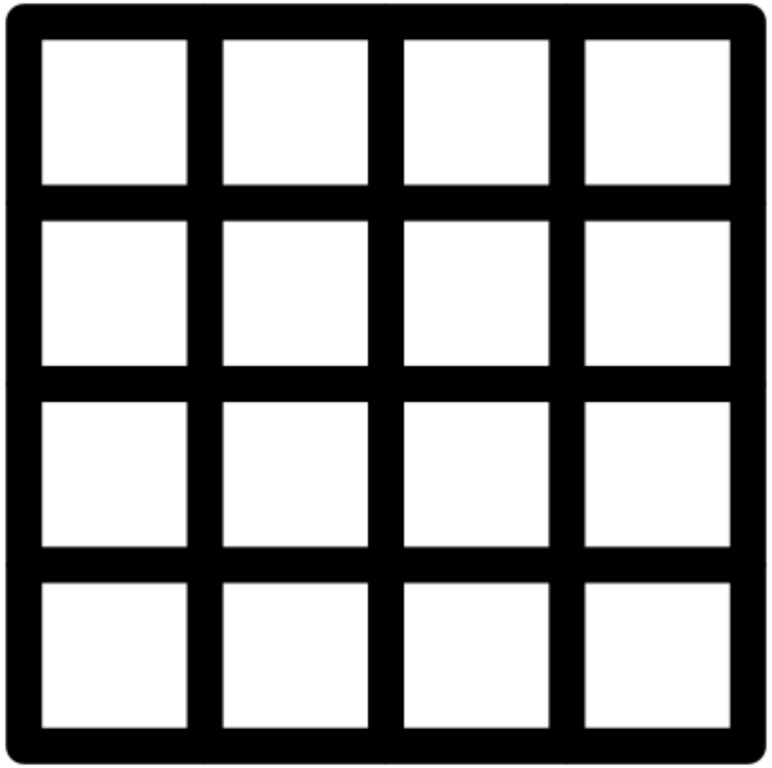
7. Number of characters in a name
8. % of numbers in name
9. % of underscores in a name

Path related:

10. File path without numbers
11. How many '/' (Depth of file)
12. Number of '.' in a path

# Approach: Hyper-parameter Tunning

## Grid Search CV



| Model | Selected Hyperparameters |
|---|---|
| KNN | 'clf__leaf_size': 10<br>'clf__n_neighbors': 3<br>'clf__p': 1,<br>'preprocessor__path__vect__max_df': 0.9<br>'preprocessor__path__vect__min_df': 0.005 |
| CART | 'clf__criterion': 'entropy'<br>'clf__max_depth': 16<br>'clf__min_samples_leaf': 1<br>'clf__min_samples_split': 2<br>preprocessor__path__vect__max_df': 0.95<br>'preprocessor__path__vect__min_df': 0.005 |
| NB | 'clf__alpha': 1.0<br>'preprocessor__path__vect__max_df': 0.9<br>'preprocessor__path__vect__min_df': 0.005 |
| SVM | 'clf__C': 10<br>'clf__gamma': 'scale'<br>'clf__kernel': 'rbf'<br>'preprocessor__path__vect__max_df': 0.9<br>'preprocessor__path__vect__min_df': 0.005 |
| RF | 'clf__bootstrap': False<br>'clf__max_depth': None<br>'clf__max_features': 'auto'<br>'clf__min_samples_leaf': 2<br>'clf__min_samples_split': 5<br>'clf__n_estimators': 100<br>'preprocessor__path__vect__max_df': 0.9<br>'preprocessor__path__vect__min_df': 0.005 |
| NN_MLP | 'clf__activation': 'tanh'<br>'clf__alpha': 0.0001<br>'clf__hidden_layer_sizes': (80, 50)<br>'clf__learning_rate': 'constant'<br>'clf__learning_rate_init': 0.001<br>'clf__solver': 'adam'<br>'preprocessor__path__vect__max_df': 0.95 |

# Approach: Performance Metrics

1. Precision (P): is the ratio of true positive (TP) predictions out of all positive predictions (TP and FP (False Positive)).

$$P = \frac{TP}{TP + FP} \qquad (1)$$

2. Recall (R): also referred to as sensitivity is the ration of TP predictions out of the actual positive items (FN is False Negative).

$$R = \frac{TP}{TP + FN} \qquad (2)$$

3. F1-Score (F1): is a harmonic mean of precision and recall, thus a good F1-Score requires a good score on both of recall and precision simultaneously.

$$F1 = \frac{2PR}{P + R} \qquad (3)$$

4. Accuracy (ACC): is the ratio of correctly classified items out of all items. (TN is True Negative).

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \qquad (4)$$

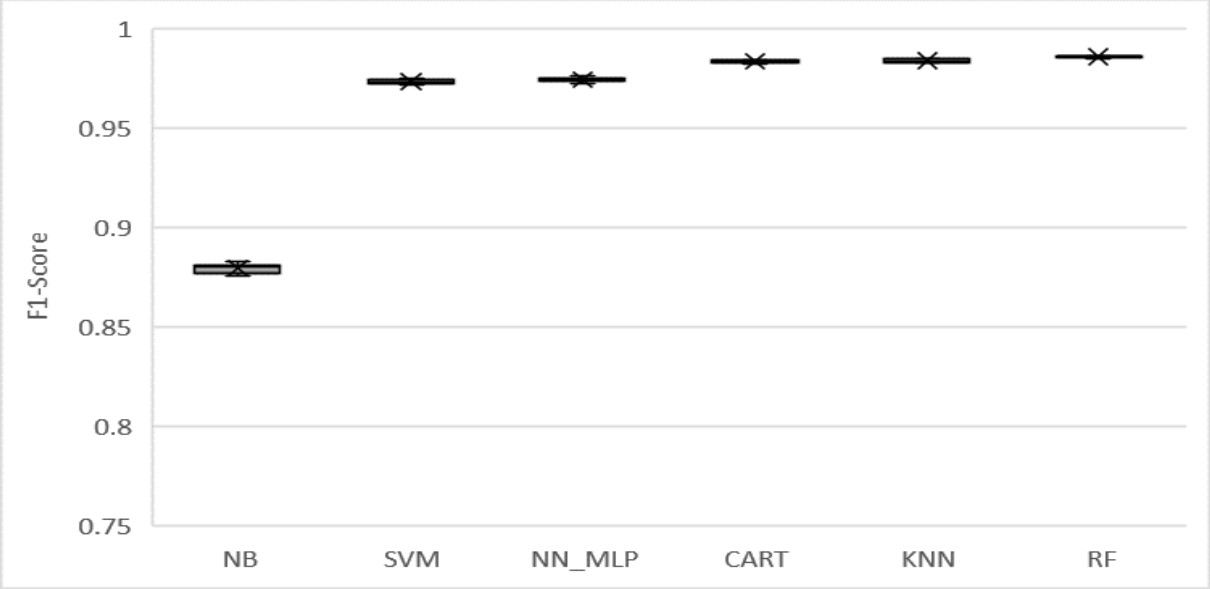
**F1 is the core evaluation metric**

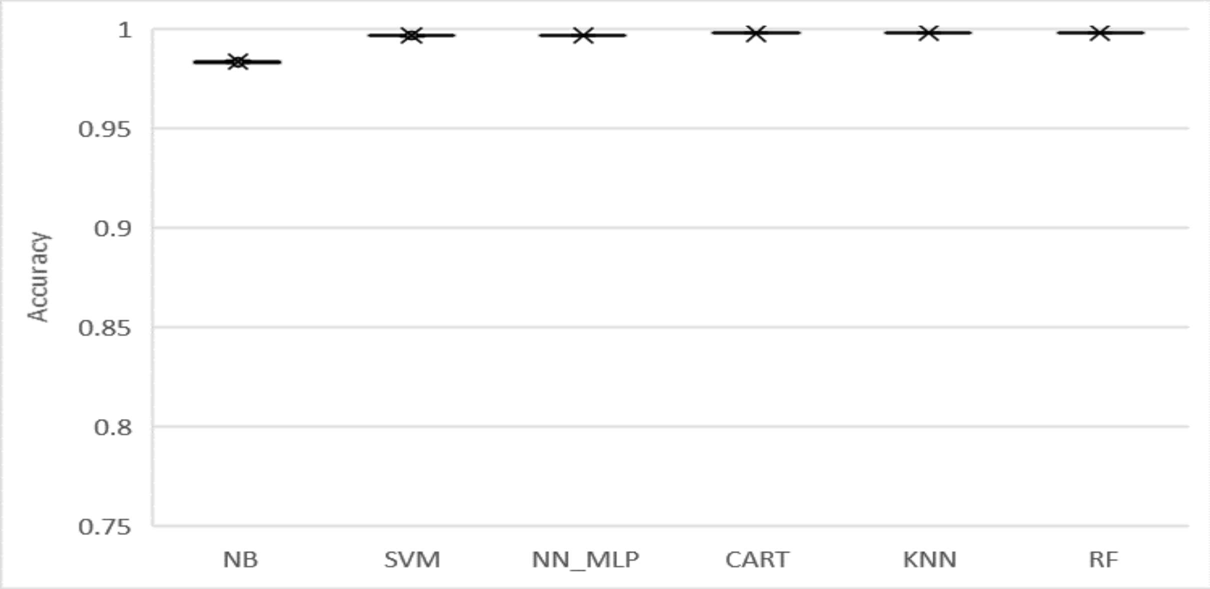

**10-fold cross validation**

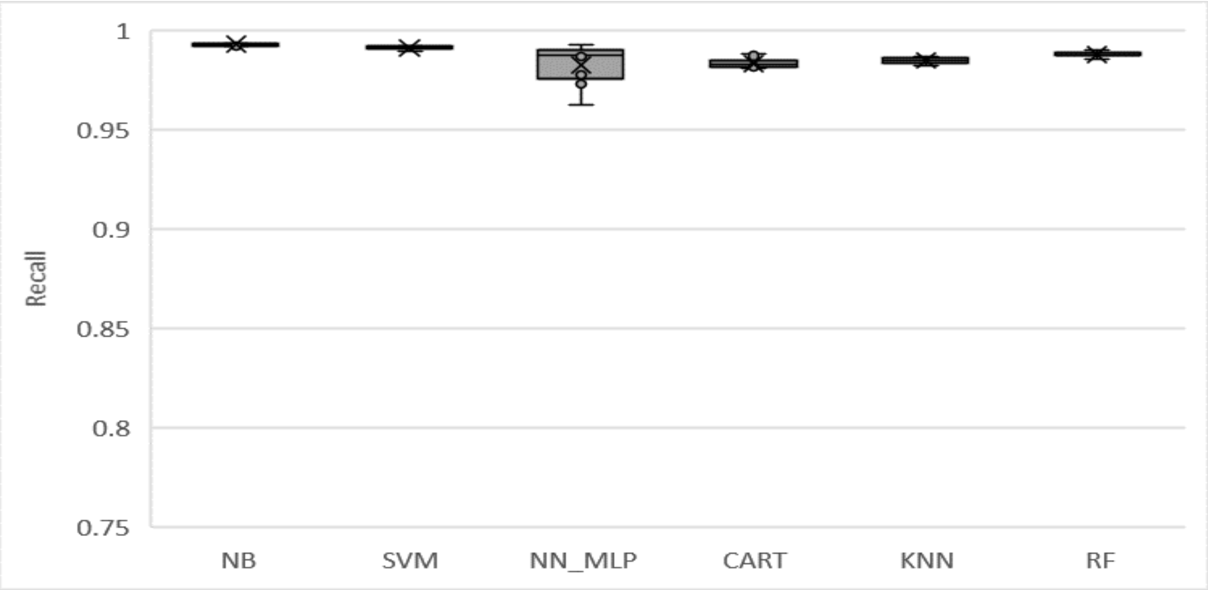
**Pipelining**

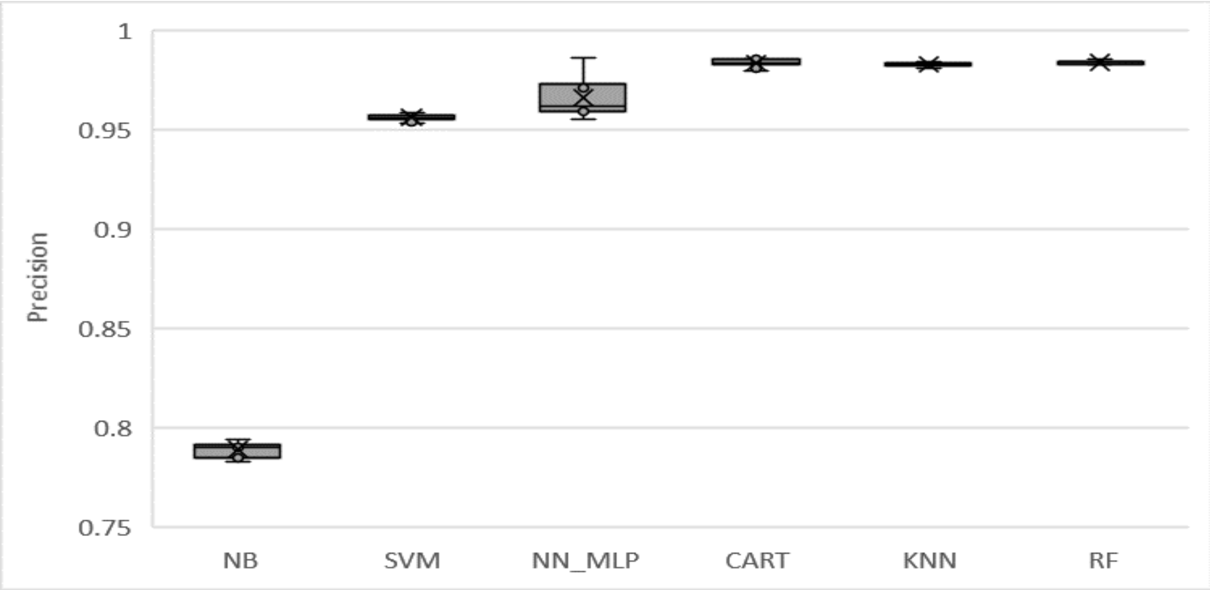
# Content

# Predictive Performance (1/2)



# Predictive Performance (2/2)

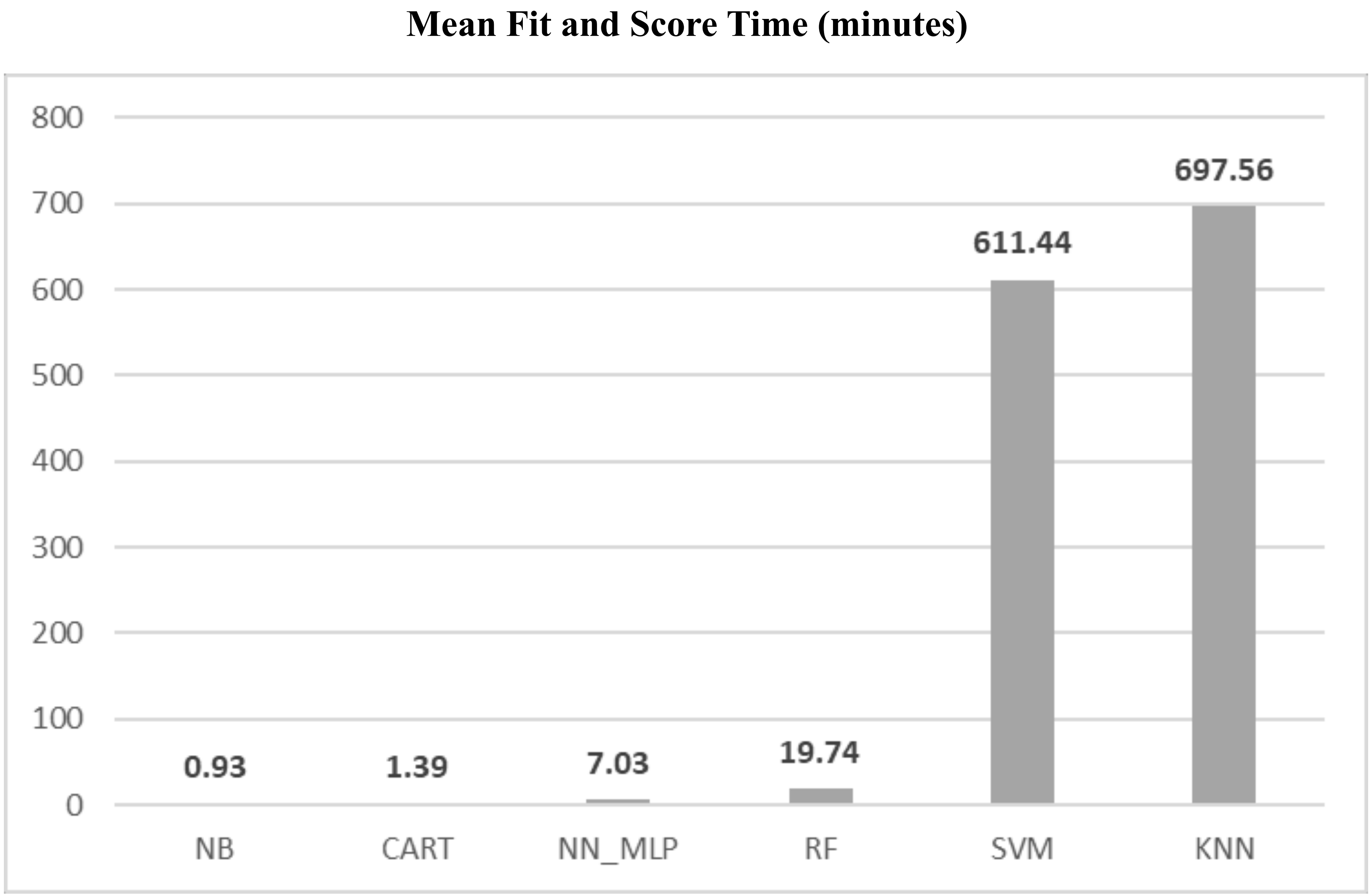
**Summary of Performance Metrics**

| Classifier | F1-Score | Precision | Recall | Accuracy |
|---|---|---|---|---|
| NB | 0.8795 | 0.7891 | **0.9932** | 0.9838 |
| SVM | 0.9735 | 0.9562 | 0.9913 | 0.9968 |
| NN_MLP | 0.9744 | 0.9664 | 0.9827 | 0.9969 |
| CART | 0.9837 | 0.9838 | 0.9835 | 0.9981 |
| KNN | 0.9840 | 0.9830 | 0.9851 | 0.9981 |
| RF | **0.9861** | **0.9841** | 0.9881 | **0.9983** |

**Summary of Standard Deviation**

| Classifier | F1-Score | Precision | Recall | Accuracy |
|---|---|---|---|---|
| NB | 0.0024 | 0.0037 | 0.0009 | 0.0004 |
| SVM | 0.0011 | 0.0017 | 0.0008 | 0.0001 |
| NN_MLP | 0.0011 | 0.0095 | 0.0098 | 0.0001 |
| CART | 0.0006 | 0.0020 | 0.0024 | 0.0001 |
| KNN | 0.0007 | 0.0010 | 0.0014 | 0.0001 |
| RF | 0.0006 | 0.0008 | 0.0012 | 0.0001 |

# Execution Time



# Content

Problem Statement

Research Objectives

Related Work

Methodology

Results

Conclusion and Future Work

# Conclusion

1. File metadata can indicate the relevance of a file for an investigation.

2. ML classification algorithms can model the decision-making process required to identify files of interest.

3. Different classification models will perform differently with RF exhibiting the best performance.

# Possible Future Work

1. Test this approach on other smart phone operating systems such as iOS as well as on other smartphone’s datasets.
2. Explore and compare the performance of deep learning classification algorithms.
3. Explore the possibility and effect of augmenting metadata features with features extracted from file content to enhance predictive performance.

# Thank you