[RFC] Rating Risk Exposure of Unstructured Database

Summary: Build a tool to allow Security analysts to understand Data Privacy exposure of a firm.

Created: May 22, 2021
Current Version: 1.0.0
Target Version: 1.1.0
PRD: Link to PRD if applicable

Status: WIP | In-Review | Approved | Obsolete

Owner: divyank.agarwal1993@gmail.com
Contributors: divyank.agarwal1993@gmail.com
Other stakeholders: divyank.agarwal1993@gmail.com
Approvers: divyank.agarwal1993@gmail.com

The RFC provides a technical solution to measure the PII risk that a database is exposed to. It provides a set of statistical measures which can be used to Label any database as Low, Medium and High Risk category.
The service should be designed to support internet level scales, i.e. ability to handle TB of data, millions of documents providing a fast turnaround time.
We should be able to scale the services to measure different criterias for eg. : phone number, Address, SSN, email address, DOB. etc.
The following solution only concerns the Unstructured data stored in NoSQL database. For Structured Data, checkout out RFC- 4289

Background

Borneo is in the business of privacy management. We reduce the data privacy risk that organizations across the world face. Borneo does this by discovering the data risk spread across the organization's data assets, Understand the type of risk and remediate the risk by taking and suggesting proactive actions.

The first step of this process is to discover the risk , i.e. provide a mathematical measure to calculate risk associated with an unstructured database. This is the context for developing a tool to calculate risk exposure with any unstructured database

Proposal

We propose a Software as a Service on which the end User can Register and login via his credentials. Once inside the Service, the end User can connect to different cloud databases via pre-built integrations.For e.g S3, Box, mongo etc. On activating **BORNEO PII RISK ASSESSOR,** our service will discover every bucket that is available for Search. We then iteratively go through each bucket and collect certain stats. Then we report Bucket level stats as well as aggregated database level stats.

Assumptions

- We assume S3 as our data source and describe our solution from S3 perspective. We don't ingest the data in our sources but run certain queries to Model the data source.

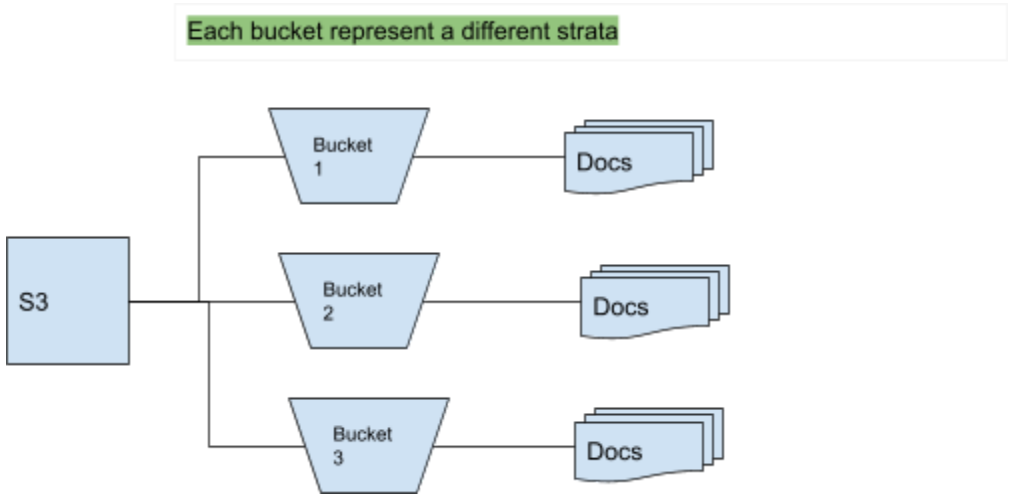
- Any database we search will contain different buckets. We assume that each bucket contains one type of data source (Strata) and they are independent of each other and contain textual data in English language.

Implementation

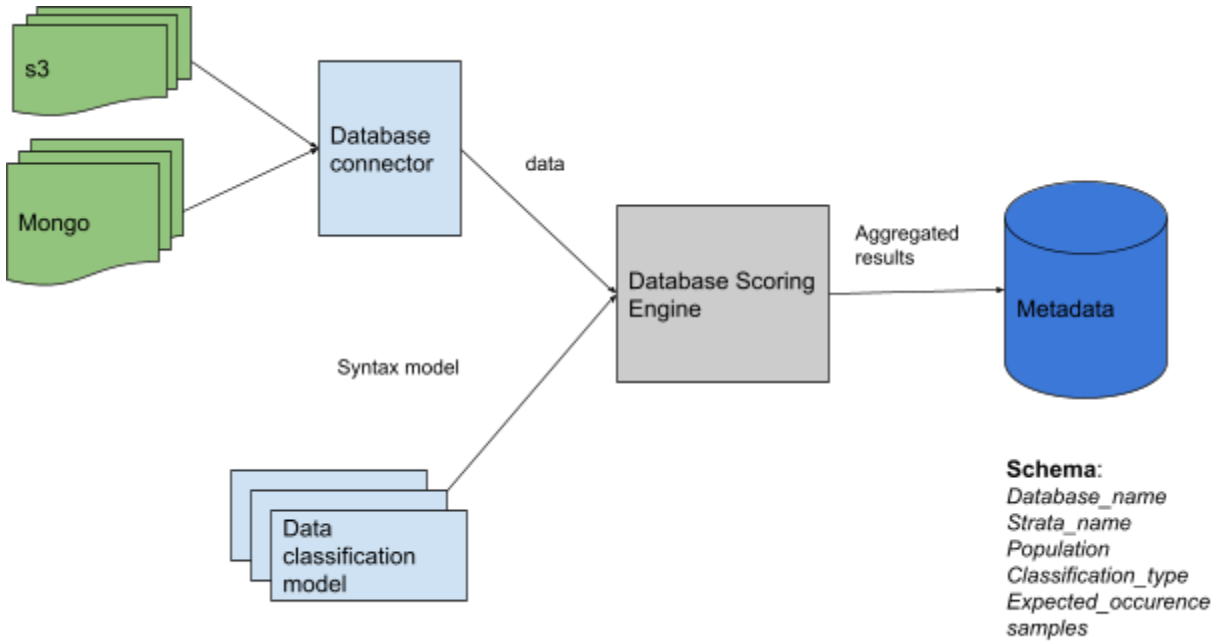
- We provide a python script which takes the database name as input . It has hardcoded pattern matching to detect US SSN in text data.

- Since we haven't built a connector to S3, we use a local file structure to create a dummy database. We provide a script to generate this database, various folders representing buckets and populate it with thousands of documents with SSN data sprinkled across the database.

Source Data Model



Architecture Diagram



Classification Data model

PII RISK :: US social security number

Social Security number is modelled as a string of Format XXX-XX-XXXX where X is any digit and the entire token is surrounding any character type.

Method 1 (implemented):

Make a regex search, with associated high probability of SSN.

Limitations:

- **False positive** : product mobile phone number 9999-00-6666 might match with the same string.
- **False Negative** - IF the SSN is stored like XXX:XX:XXXX, we might not match it.

Method 2:

Steps:

- Tokenize the entire document
- Build features for each token
- Use a trained Decision Tree model to associate SSN probability score with each token

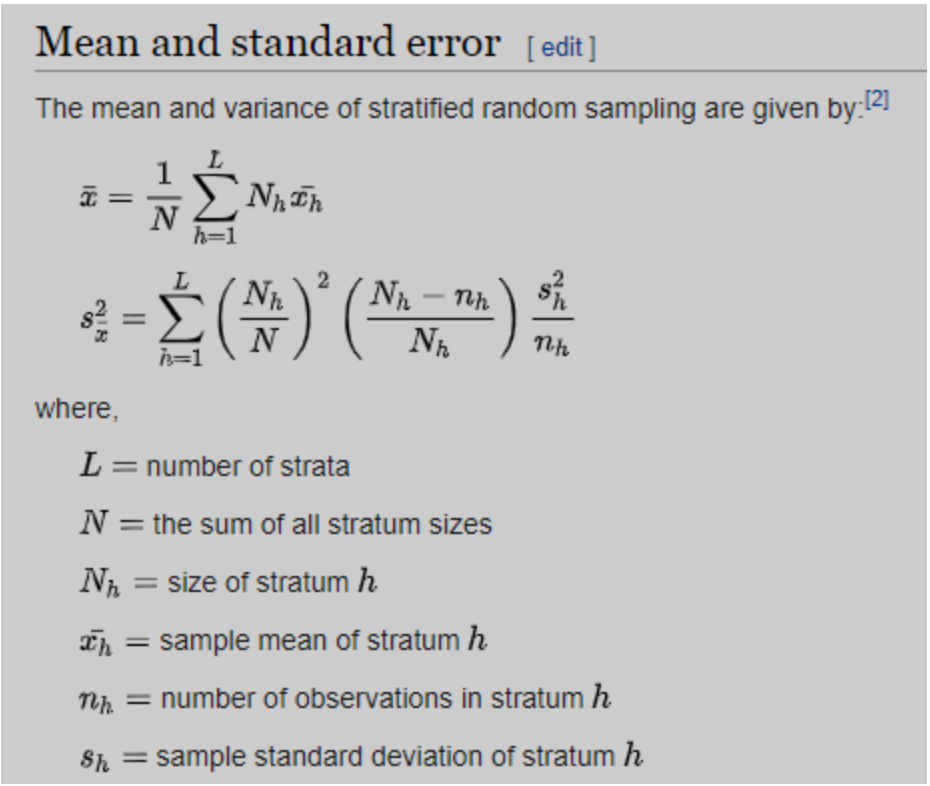
How to train Decision Tree model

- Collect training token
- Build features for each Training Token
    - Regex match to \d{3}-\d{2}-\d{4}
    - Total length of Token
    - Total digits in token
    - Total Alphabets in token
    - any_mention_of_SSN_in_document
    - Char_immediately_before_match_is_digit
    - Char_immediately_before_match_is_alphabet
    - Char_immediately_after_match_is_digit
- Associate ground truth to each Token
- Using scikit-learn to train the model.

Sampling Algorithm

- For buckets which have less than 50 documents, we don't sample and go through all 100 documents. Otherwise we follow proportionate allocation Stratified Sampling.

- We treat each bucket independently and sample xx% of its data based on simple random sampling. We then aggregate the results for the database as Stratified sampling with each strata as a bucket.
- Sampling is only restricted to sampling documents, we dont sample text content within a document.



Mean and standard error [edit]

The mean and variance of stratified random sampling are given by:[2]

$$\bar{x} = \frac{1}{N}\sum_{h=1}^{L} N_h \bar{x_h}$$

$$s_{\bar{x}}^2 = \sum_{h=1}^{L} \left(\frac{N_h}{N}\right)^2 \left(\frac{N_h - n_h}{N_h}\right) \frac{s_h^2}{n_h}$$

where,

L = number of strata

N = the sum of all stratum sizes

N_h = size of stratum h

$\bar{x_h}$ = sample mean of stratum h

n_h = number of observations in stratum h

s_h = sample standard deviation of stratum h

Source : *https://en.wikipedia.org/wiki/Stratified_sampling*

Simple Random Sampling :

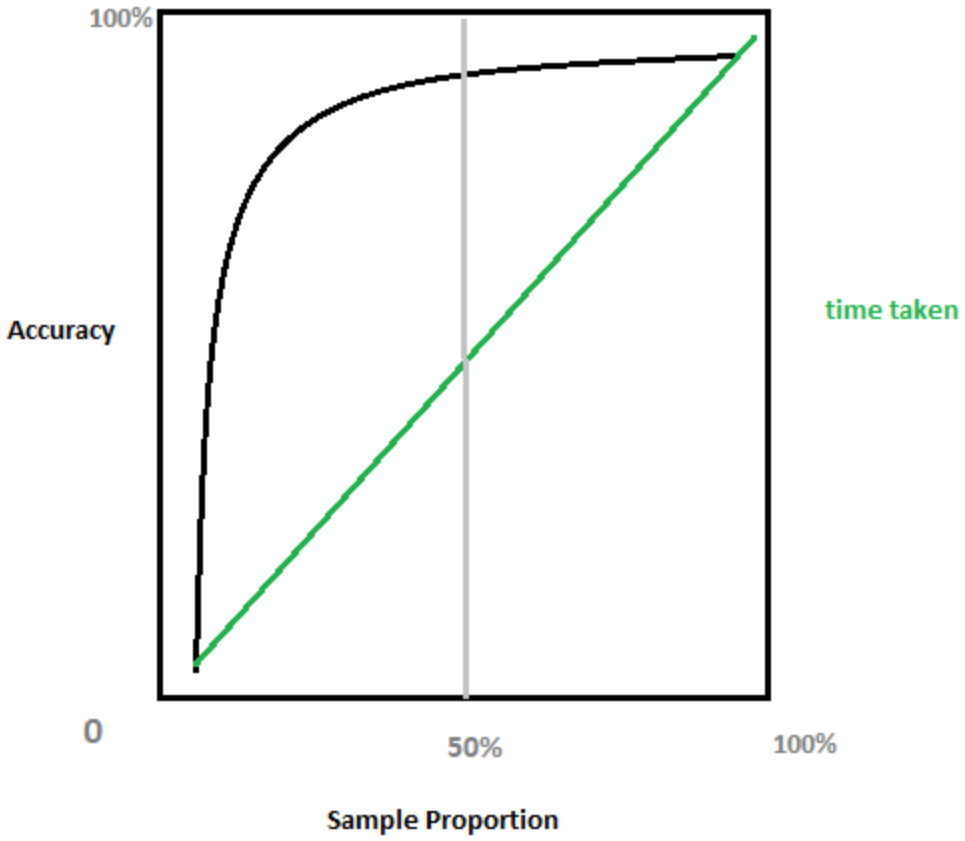
In simple random Sampling , each individual is chosen randomly and is entirely by chance. I.e probability of selecting any individual is the same.

Random Sampling:

The probability differs among different choices/ Strata.

Trade OFFs

Time taken Vs Accuracy Vs sample proportion



- Accuracy increase is significant around 10% sample proportion, but after that increase in Accuracy benefit is not substantial as compared to increase in sampling cost.

Sample Size Vs Precision

Precision is a measurement is depends upon the length of confidence interval
The larger the interval the lower is the precision.
Hence, presion is inversely proportional to confidence Interval range

precision $\propto \sqrt{n}$
when a normal distribution is followed.