OPENSTACK LOG DATA ANALYSIS

• RAVI CHARAN

PROBLEM STATEMENT

- Massive amount of data is generated by the Openstack cloud services in the format of service logs.
- The logs contain information that is useful for patten analysis.
- Unfortunately this information is generally exposed in semi-structured text format, not allowing direct analysis without additional munging of the data.

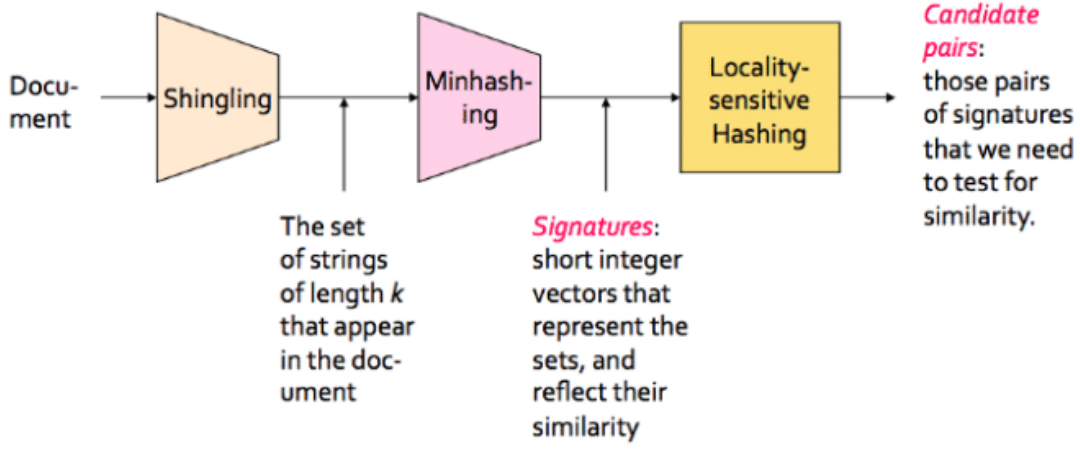
TRADITIONAL APPROACH

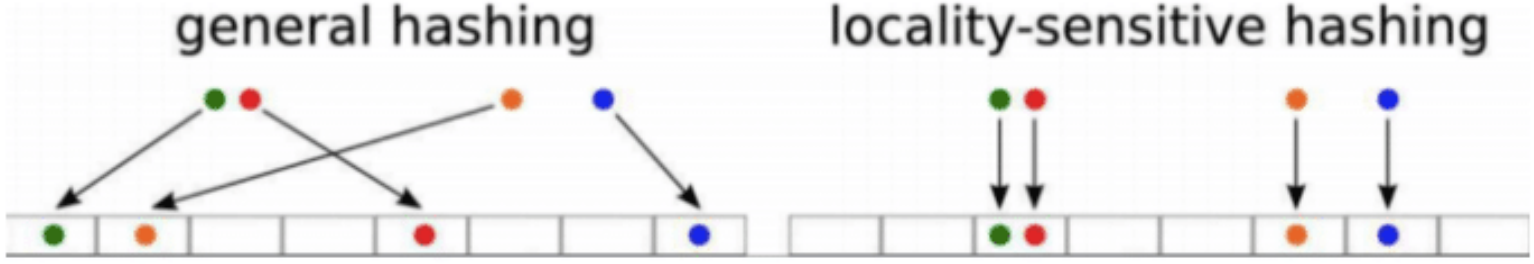
- Rule Based : Usage of Regular Expressions to extract common patterns.
- This approach requires a pre-knowledge of all text patterns and are not scalable with the services growth.
- Neural Nets : Computationally complex since the entries are approximately 1.5*10^6/per hour.

LOCALITY SENSITIVE HASHING

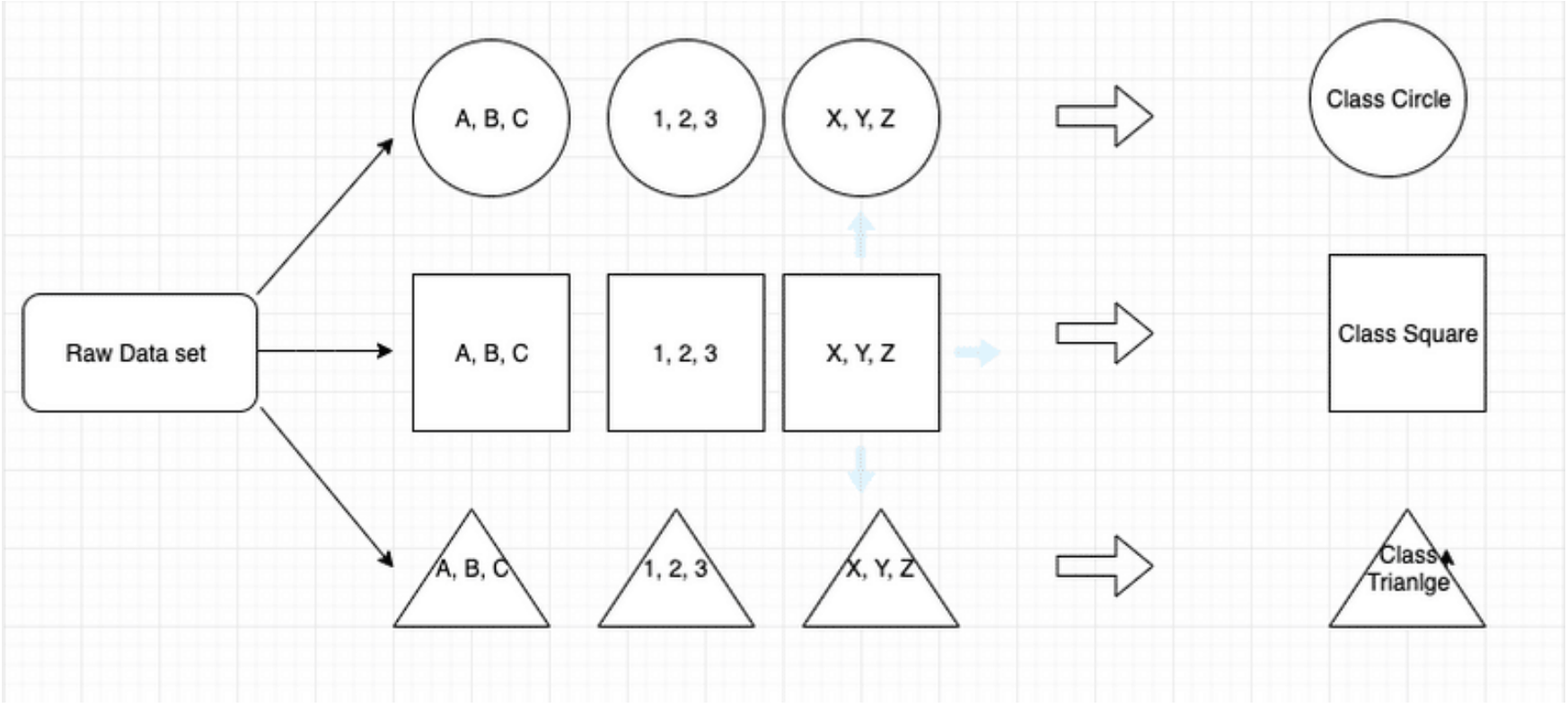
- Majorly finds it's application in pattern analysis by tech giants, analyzing RNA samples, and ***near duplicate/similar detection in documents.***

THE ALGORITHMIC APPROACH



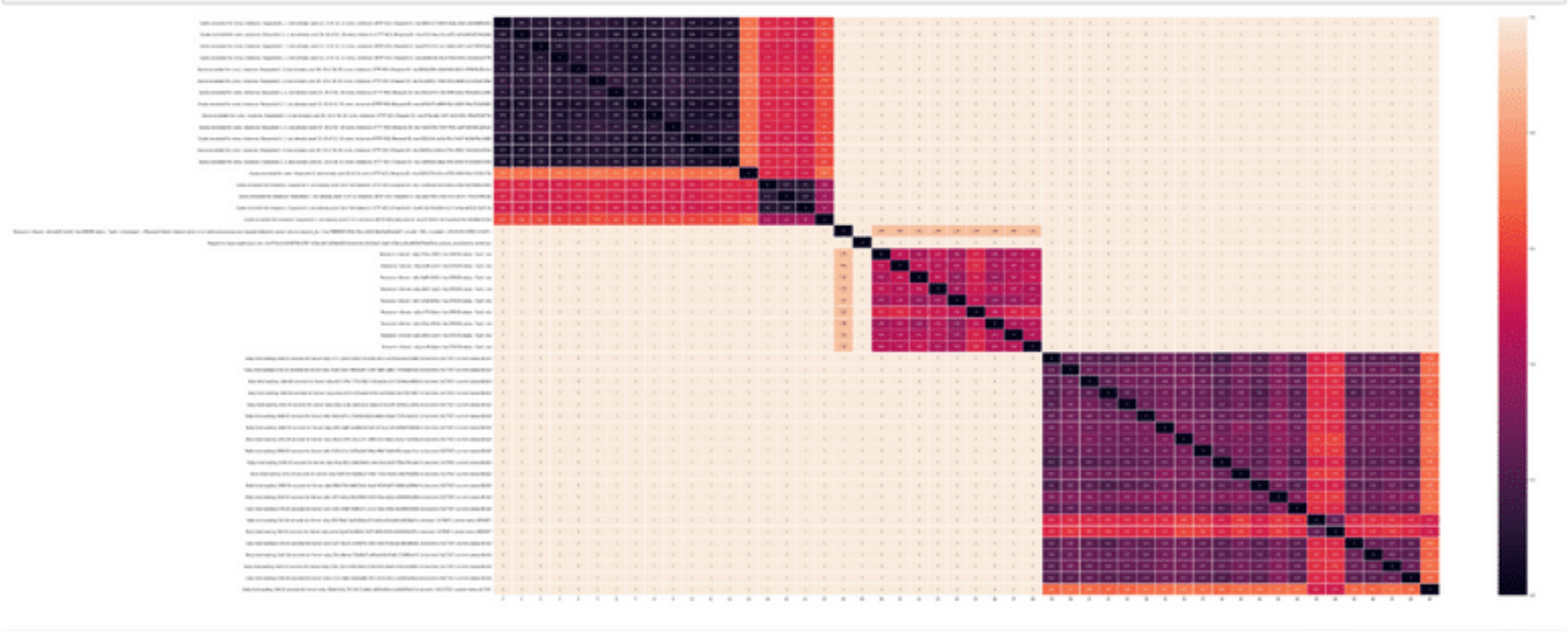


CERN openlab



RawData -> Clustering -> Classification -> Anomaly Detection

EXAMPLES



CONCLUSION

- Extend the clustered data for ***anomaly detection.***
- Usage of genetic algorithms to generate regular expression from every cluster and mark them as the principal component.
- Jupyter notebooks: https://github.com/RavicharanN/Rally-Log-data-Analysis
- Project report: https://github.com/RavicharanN/Log-Data-Analysis-Report

THANK YOU!

Supervised by:

- DOMENICO GIORDANO
- JOSE CASTRO LEON
- SPYRIDON TRIGAZI

THE JACCARD INDEX

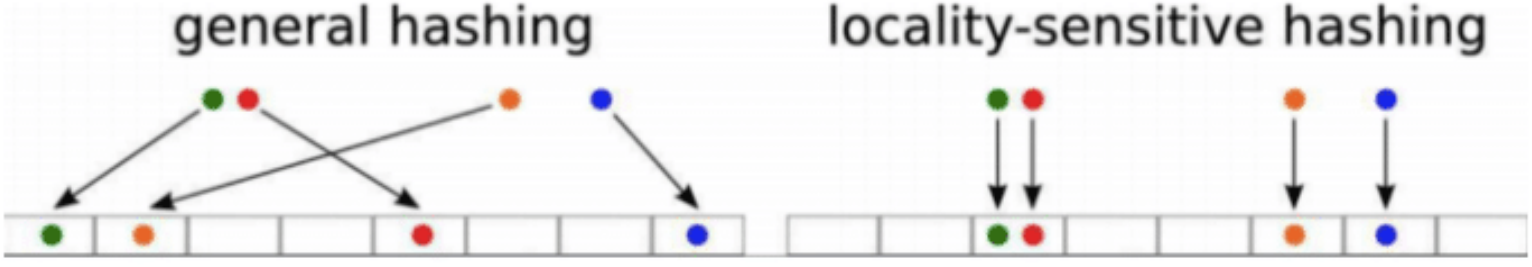
- Jaccard index is the metric used to compare the similarity between two messaged. Trivially, its given by :
  - *J(A, B) = |A^B| / |AUB|*

- Message1 = {"AA BB CC"} Message2 = {"CC DD EE"}
  - *Similarity = 1/5*

- ***Shingling:*** This is the technique that is used to preserve the order of the tokens in the messages as Jaccard similarity doesn't take into account the order of the tokens. We use a 2-shingle approach, which is, we consider 2 consecutive tokens as one word.

- **Minhash:** We generate small integers signatures for the set of shingles which reflect their similarties. These signatures are small enough to fit in the memory and this reduce the face complexity.

- ***Locality Sensitive Hashing:*** The general idea of LSH is to find a algorithm such that if we input signatures of 2 documents, it tells us that those 2 messages form a candidate pair. It is a method in which hash data points into buckets so that data points near each other are located in the same buckets with high probability



DATA ANALYTIC SOLUTION

- Since the logs are always generated by the machine, the number of tokens (words) for all the messages in a given class are same.

	atime	task	deployment	raw	dtime	msg	_info	word_count
0	1524387149611	attach-volume	wig_project_003	2018-04-22 10:52:29.611 17979 ERROR rallytester.rallytester [-] [nova attach-volume wig_project_003] Task failed: Rally tired waiting 1440.00 seconds for Server rally-9db3-aPln:176b6fd3-8a2d-4b60-a5bd-73575c4ad161 to become ('ACTIVE') current status BUILD	2018-04-22 08:00:00	waiting for Server to become ('ACTIVE')	Rally tired waiting 1440.00 seconds for Server rally-9db3-aPln:176b6fd3-8a2d-4b60-a5bd-73575c4ad161 to become ('ACTIVE') current status BUILD	14
3	1524387321429	boot-linux	gva_project_013	2018-04-22 10:55:21.429 19322 ERROR rallytester.rallytester [-] [nova boot-linux gva_project_013] Task failed: Rally tired waiting 1440.00 seconds for Server rally-15fc-N4mw:77640b75-df94-4e6d-91d6-3338fffcdc91 to become ('ACTIVE') current status BUILD	2018-04-22 08:00:00	waiting for Server to become ('ACTIVE')	Rally tired waiting 1440.00 seconds for Server rally-15fc-N4mw:77640b75-df94-4e6d-91d6-3338fffcdc91 to become ('ACTIVE') current status BUILD	14

SAMPLE SPACE REDUCTION

- The first step in the top-down approach of the clustering algorithm is the dividing the dataset on the basis of the number of tokens.

- Run Minhash-LSH on each of the reduced clusters to further divide them into smaller clusters

VIOLIN SEABORN DISTRIBUTION