SimCorp

Finding Common Root Causes in Autotest Results

(Applied Machine Learning techniques)

Allan P. Engsig-Karup[1], **Steffen Holmslykke**[1], **Søren Lindegaard Grubov**[2]

Introduction

As a software provider, SimCorp needs to handle a large amount of failing autotests across many installations world-wide during an implementation period. Such failing auto tests of SimCorp Dimension installations are handled manually and individually every day as a part of continuous quality assurance. Even though the tests are different, many of these tests fail with a common root cause due to a shared flow. If the failed tests with the same root cause can be grouped together, the amount of duplicate work could be reduced and the number of uniquely handled root causes could be increased leading to better resources spent on handling such errors.

Research Question

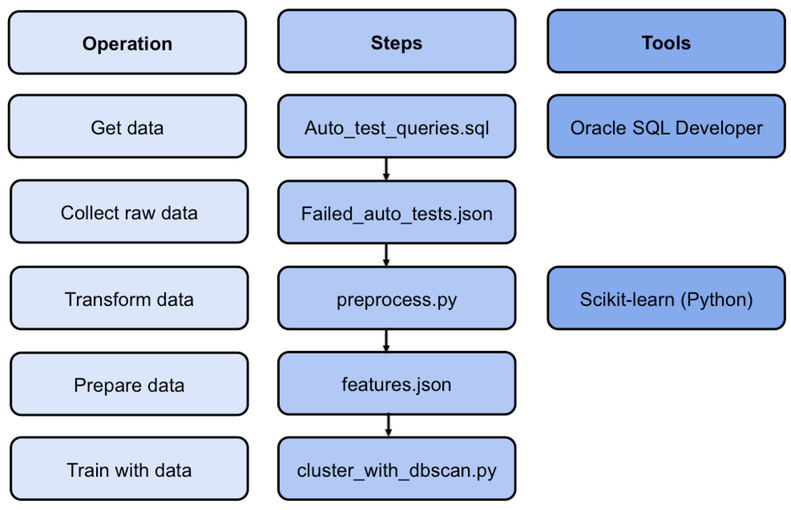
Can we automatically detect different types of errors in the Auto Test results and group these together?

Problem Setting

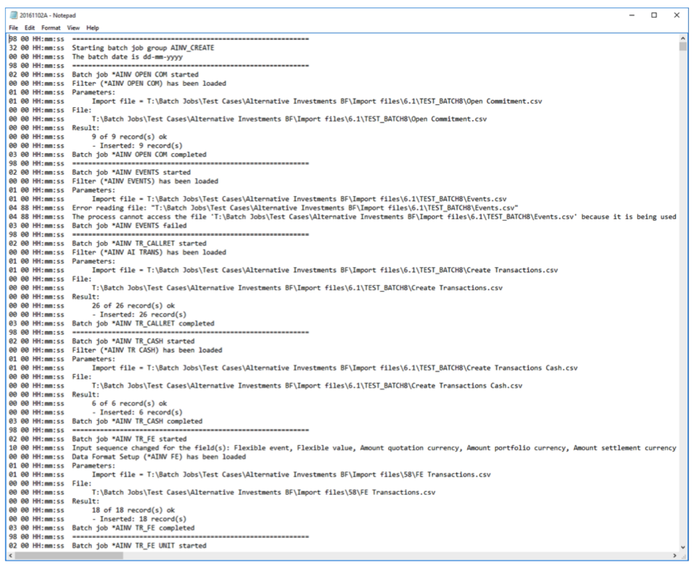
Daily error logs are produced across all SimCorp Dimension installations. These errors logs are simple text files that contains information from the batch jobs that are executed to test the installations. These logs contains information about the execution and the errors that have resulted. However, the errors reported are described with textual information that may not be uniquely classified immediately, and therefore it can be difficult to describe root cause without warranting further investigations into the code. For example, it may be that due to one test failing, then subsequently a number of other tests also fails and in such cases the root cause is the first test that failed. Also, it is known how the correct output should look like for a given batch job that are executed successfully and therefore it is possible to identity the textual information about each error that have resulted.

Machine Learning Workflow

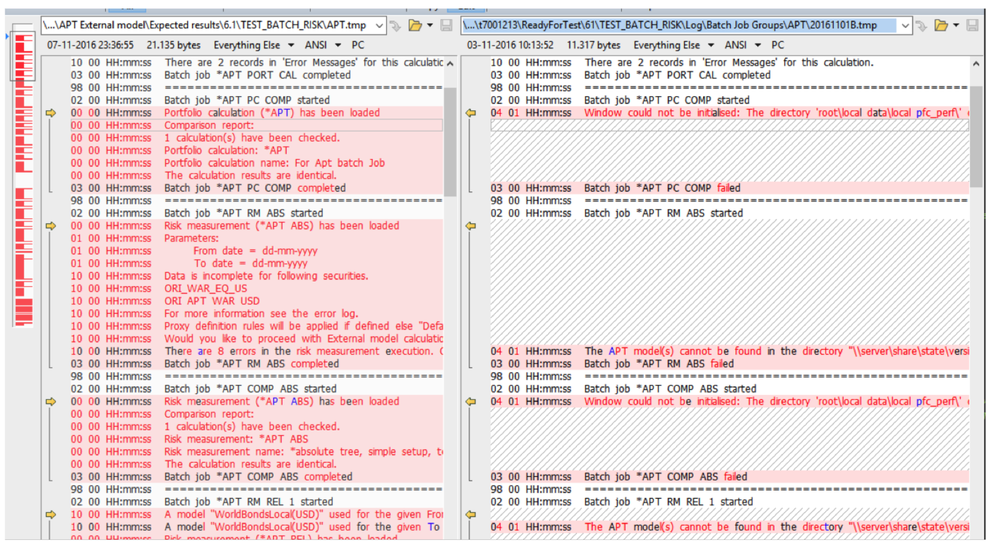
We employ the scientific python framework SciKit-Learn [2] to process the raw data and transform it into a structured data set using a bag of words model for identifying patterns in the error messages.



Error Log Data



Differences between error log data and expected log data

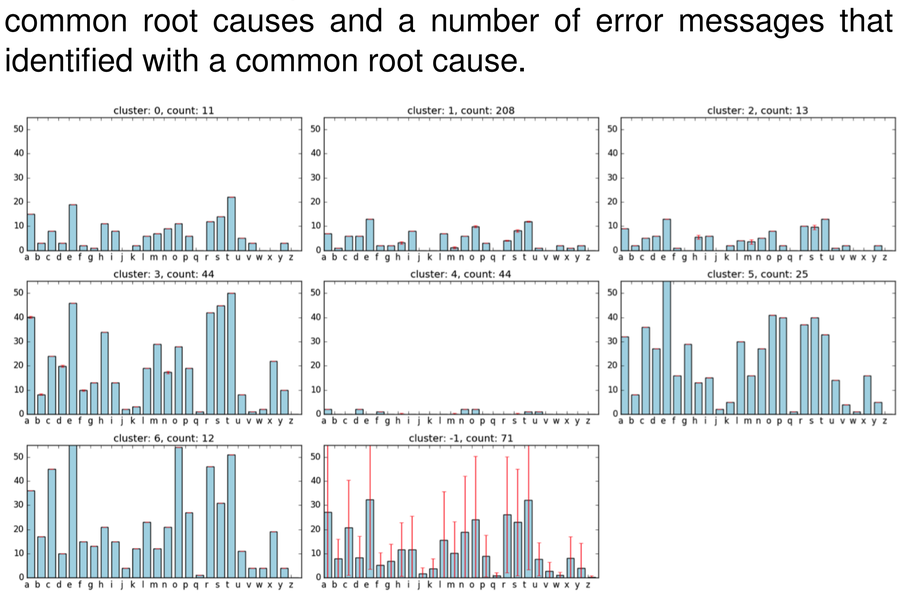


Feature Engineering

To define a feature vector that can be used for training using DBSCAN [1] to automatically detect clusters, we can use a Bag of Words model / N-gram model. The chosen feature set is here defined in terms of letters (unigrams), from which we can determine the frequency count of each letter and use the resulting vector as a 'fingerprint' of the error messages.

Results of Analysis

The results of the unsupervised analysis using DBSCAN on the normalised structured dataset based on sparse feature vectors, shows that we can identify key clusters of common root causes and a number of error messages that cannot immediately be identified with a common root cause.



References

[1] Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. AAAI Press, pp. 226–231.

[2] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research 12* (2011), 2825–2830.

Legal Disclaimer

* **Corresponding author: aptk@simcorp.com**, [1] SimCorp Technology Labs, SimCorp A/S, Denmark, [2] SimCorp A/S, Denmark.