# ML4DevOps.

Table of Contents

[ML4DevOps. 1](#_Toc7203144)

[Background. 1](#_Toc7203145)

[1. Case Studies ML for DevOps. 1](#_Toc7203146)

[1.1. Tracking Application Delivery 1](#_Toc7203147)

[1.2. Ensuring Application Quality 2](#_Toc7203148)

[1.3. Securing Application Delivery 2](#_Toc7203149)

[1.4. Managing Production 2](#_Toc7203150)

[1.5. Managing Alert Storms 2](#_Toc7203151)

[1.6. Troubleshooting and Triage Analytics 2](#_Toc7203152)

[1.7. Preventing Production Failures 3](#_Toc7203153)

[1.8. Analyzing Business Impact 3](#_Toc7203154)

[2. ML for Log Analytics. 3](#_Toc7203155)

[2.1. Basic concept. 3](#_Toc7203156)

[2.2. ML for detecting system failures 4](#_Toc7203157)

[3. Anomaly detection using time series data. 4](#_Toc7203158)

[Step 1. Collecting data. 4](#_Toc7203159)

[Step 2. Python notebook: 4](#_Toc7203160)

[Resources and links. 5](#_Toc7203161)

[1. Pull all kinds of necessary data from a variety of sources 5](#_Toc7203162)

[2. Extract, transform, and load the data 5](#_Toc7203163)

[3. Avoid technical and performance drawbacks when everything ends up in “one big table” at the end 5](#_Toc7203164)

[4. Facilitate continuous machine learning and decision-making in a business-ready framework 5](#_Toc7203165)

## Background.

## 1. Case Studies ML for DevOps.

Some of the key examples of applying Machine Learning to DevOps include:

### 1.1. Tracking Application Delivery

The activity data from ‘DevOps tools’ such as Git, SonarQube, Jira, Ansible and many others provide delivery process visibility. Application of ML on these tools uncover the anomalies in that data- substantial code volumes, long build times, late code check-ins, slow release rates to identify many of the software development wastes, including excessive task switching, gold plating, inefficient resourcing, partial work, or process slowdowns.

### 1.2. Ensuring Application Quality

ML, by analyzing output from testing tools can intelligently review QA results, efficiently build a test pattern library based on discovery. This understanding of a ‘known good release’ helps to ensure comprehensive testing on every release, even for novel defects, increasing the quality of delivered applications.

### 1.3. Securing Application Delivery

Like fingerprints, User behavior patterns can be unique. Applying Machine Learning to Dev and Ops user behaviors helps in identifying anomalies, which represents dangerous activity. *For instance*, the access of anomalous patterns to repos, deployment activity, automation routines, test execution, system provision, and more can highlight users were exercising ‘familiar bad patterns’ in a rapid pace both intentionally or accidentally. These patterns include deploying unauthorized code, coding back doors, stealing intellectual property, etc.

### 1.4. Managing Production

Machine Learning comes into its own by analyzing an application in production, due to larger data volumes, transactions, etc. occurs in prod when compared to dev or test. The teams of DevOps use ML to analyze general patterns including resource utilization, user volumes, etc. and finally to detect abnormal patterns like memory leaks, DDOS conditions, and race conditions.

### 1.5. Managing Alert Storms

The practical and best-value application of ML is in managing the gigantic flood of alerts, which occurs in the systems of production. It can be more complicated such as “training systems overtime to identify ‘known well’ and inadequate warnings, thus enabling filtering to reduce alert storms and fatigue.

### 1.6. Troubleshooting and Triage Analytics

The other area where Ml technologies shine today is in triage analytics. It can automatically detect and can triage known issues and some unknown ones even. These tools can detect anomalies in general processing and can analyze the release logs to correlate with new deployments. Even other automation tools can use ML to raise a ticket, alert operations and assign them to the exact source.

### 1.7. Preventing Production Failures

In the prevention of failures, ML can go beyond the straight-line capacity planning. It can map utilizing patterns to predict. The required configuration for a desired level of performance, the percentage of clients can use a brand new feature, infrastructure necessities for a brand new promotion, an outage can impact the engagement of customers. ML sees opaque early indicators in applications and systems, allowing Ops to avoid problems faster with quick response times.

### 1.8. Analyzing Business Impact

In DevOps, to achieve success, understanding the impact of code release on business goals is critical. ML systems can detect good and bad patterns by analyzing the metrics of the user thus generate an early warning system to business teams and coders when a problem arises in applications.

### 

## 2. ML for Log Analytics.

### 2.1. Basic concept.

The basic concept of Machine Learning usage for log [analytics](https://www.upwork.com/hiring/for-clients/analytics-drive-business/) can be explained with an example. Three types of inputs are obtained:

* system counters, CPU, memory, disk, and network.
* distributed logs from different applications around your system.
* error logs, crashing of executable programs, improper shutting down of applications etc.

After the collection of all these input sources, a relevant type of information from these logs is extracted automatically with the use of Bayesian Algorithm. Relevant logs are obtained as an output. Machine learning is used to aggregate the logs automatically into correlated categories. Then, newly log data will automatically incorporate into the corresponding category.

2.2. ML for detecting system failures.

* Select training data as the representation of features of log data and use them to fit the appropriate model according to the given dataset. Training data is used to recognize the failure within the system.
* evaluate the performance of the model using test dataset. This is the process of supervised learning i.e. log data patterns can be defined in advance.
* If log data patterns cannot be defined in advance unsupervised learning is introduced. In this approach, most relevant patterns are taken without the need of training dataset provided by the human being.

## 3. Anomaly detection using time series data.

Anomaly detection problem for time series is usually formulated as *finding outlier data points relative to some standard or usual signal*. Examples of anomaly types : unexpected spikes, drops, trend changes and level shifts.

An anomaly detection algorithm types :

* labeling each time point with *anomaly/not anomaly*
* forecasting a signal for some point and test if this point value varies from the forecasted enough to deem it as an anomaly.

### Step 1. Collecting data.

Prometheus, a [Cloud Native Computing Foundation](https://cncf.io/) project, is a systems and service monitoring system. It collects metrics from configured targets at given intervals, evaluates rule expressions, displays the results, and can trigger alerts if some condition is observed to be true.

### Step 2. Python notebook:

Example : https://github.com/TristanCacqueray/anomaly-detection-workshop-opendev/blob/master/datasets/notebook/anomaly-detection-with-scikit-learn.ipynb

**Algorithms selection for the POC**

## 4. Next steps.

## Evaluate the “seasonality” factors in collected time series data such as date /time, user peak activities .

## Establish the correlation with the event log , calculate the confidence in predicting “good’ vs ” bad” anomalies.

## Resources and links.

* Prometheus . <https://prometheus.io/>
* <https://www.upwork.com/hiring/for-clients/log-analytics-deep-learning-machine-learning/>
* https://blog.statsbot.co/time-series-anomaly-detection-algorithms-1cef5519aef2

Raw data pre-processing

Involves following steps:

### 1. ****Pull all kinds of necessary data from a variety of sources****

a. Internal data sources like ERP, CRM, POS systems, or data from online e-commerce platforms

b. External data, like weather, public holidays, Google trends etc.

[TBD] Add data source of trace/log data for Devops

### 2.****Extract, transform, and load the data****

a. Relate and join the data sources

b. Aggregate and transform the data

[ TBD ] add tools available to devops

### 3. Avoid technical and performance drawbacks when everything ends up in “one big table” at the end

### 4. Facilitate continuous machine learning and decision-making in a business-ready framework

Utilize historic data to train the machine learning models and algorithms

b. Use the current, up-to-date data for decision-making

c. Export back the resulting decisions/recommendations to review by business stakeholders, either back into the ERP system or some other data warehouse

Our main focus for this project is on the step 4 .