# Machine Learning Engineer Nanodegree

## Capstone Proposal

Sergey Sergeev
February 4th, 2019

## Proposal

Virtual Machine (VM) preparation time prediction

### Domain Background

I work for the software development company, and we pay high attention to the test automation. During 15 years of our product development and many implemented customer projects we accumulated a lot of autotests. Autotests check various parts of our product in different environments, e.g.

- 1, 2, 4, 8 VM clusters
- RedHat Enterprise Linux (RHEL) version 6 or 7
- different versions of Java (our primary programming language)
- etc

Individual test's execution times vary from few minutes to several hours depending on complexity and size.

We built the private cloud (based on Openstack) for the continuous autotest execution and a simple Web interface to manage it. On the dedicated page test engineer or developer may choose the list of tests he want to execute, number of virtual machines, versions of the 3rdparty software to be provisioned in the VMs, and so on. After that the execution request is queued to be executed as soon as possible according to cloud capacity and current VM consumption. As a result, engineer receives an email with the test report. Test reports are also stored in some shared directory for reference, comparison and further analysis needs.

### Problem Statement

For the higher cloud capacity utilization and better user experience we would like to have a model capable to predict overall test execution time. This time is a sum of three:

- Queue time (t1): time spent by an execution request in the queue waiting for the free cloud capacity
- VM preparation time (t2): time to create a cluster of VMs and provision it with the requested 3rdparty software
- Actual test execution time (t3): actual test execution till the final report

In this project, I would focus on t2 estimation only, leaving t1 and t3 for a future.

### Datasets and Inputs

As an input, I would use historical data of 1200 VM preparations requests collected over 2 months.

- Requests : 1200 | 5496 VMs
- Daily : 17.1 | 78.5 VMs
- Success : 1192 | 99.33%
- Failure : 8 | 0.66%

For a single request, I have the following data:

- Start timestamp
- End timestamp
- Number of VMs
- VM flavor (# of CPUs, RAM and Disk)
- Base OS version: RHEL 6.9, 7.3 or 7.4
- Requested 3rdparty software (with version numbers)
- Success/Failure indication
- Number of attempts performed to create and provision VMs

### Solution Statement

The solution to the problem is a regression model developed with the help of scikit-learn library. The model should get expected VM cluster details and produce a single value of expected time period to get such cluster up and running.

## Benchmark Model

As a benchmark, I would use a simple model which doesn't take into account anything except requested VM cluster size (no details about 3rdparty software to be provisioned). The model should calculate expected time as an average over historical data for clusters of the same size (number of VMs).

## Evaluation Metrics

R^2 (coefficient of determination) regression score would be used as evaluation metric: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html

## Project Design

Here is the brief workflow to prepare the solution as I see it now:

- Clean up data: remove records about failed cluster creations - they are just garbage (0,66% of whole dataset)
- Prepare benchmark model for final solution evaluation
- Analyse and remove data outliers
- Remove useless features - e.g. success/failure indicator, number of attempts performed
- Transform original features: one-hot encode features containing software version numbers
- Consider transformation of "Number of VMs" to (0,1) range
- Try grid search cross-validation with DecisionTreeRegressor (from scikit-learn)
- Probably try other regressors for scikit-learn

(Various steps above should be accompanied with required visualizations)

- Test final model on a testing set, compare with the benchmark model.