EP2300 Project - Network Management
Estimating Conformance to Service Level Agreements (SLAs) using Machine Learning

Rolf Stadler Rafael Pasquini

October 8, 2017

Overview

Telecom and Internet services are vital to our daily life. Such services involve large and complex software systems that increasingly run on general-purpose platforms and operating systems. For the proper function of those services, a service provider performs real-time service assurance, which ensures the service-level agreement (SLA) between the provider and a customer. In this project, we study the quality of service of a video-on-demand (VoD) service, and the SLA towards a customer is expressed as the minimum number of video frames per second on the customer terminal. The project focuses on a critical part of service assurance, namely, on the capability of a provider to estimate the service quality based on measurements on the provider's infrastructure. The estimation uses machine-learning techniques, specifically regression and classification.

Project Objective and Approach

The objective of this project is to investigate SLA conformance of a video-on-demand (VoD) service using a service metric Y on the client side and device statistics X on the server side. Figure 1 gives the basic configuration of the system we consider [1]. It consists of a client machine that is connected to a server via a network. The client accesses the VoD service that runs on the server. In this setting, device statistics refer to operating-system metrics on the server side, while service metrics Y refer to statistics on the client side. Table 1 describes the metrics X and Y used in the project.

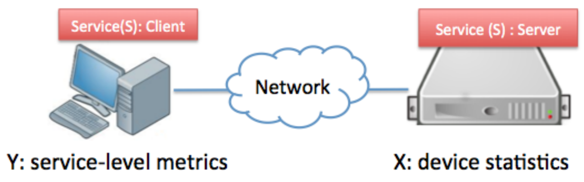

Figure 1: System configuration for estimating service metrics.

Using machine-learning techniques, the problem is to find a function (i.e., a learning model) $M : X \rightarrow \hat{Y}$, such that $\hat{Y}$ closely approximates Y for a given X. If Y is a real number, the problem is referred to as a regression problem; if Y is a label, the problem is called a classification problem. This project considers both of these problems. To estimate M, measurement pairs (or observations) of the form (X,Y) are needed. A set of measurement pairs is also called a trace, and we use in this project a trace of 3600 observations, collected once every second from running the system over the course of an hour (periodic trace in[2]).

Field ID	Description
all_..idle	Percentage of idle CPU time (%)
X..memused	Percentage of used memory (%)
proc.s	Rate of process creation
cswch.s	Rate of context switching
file.nr	Number of used file handles
sum_intr.s	Rate of interrupts
ldavg.1	Load average for the last minute
tcpsck	Number of used TCP sockets
pgfree.s	Rate of freeing pages

(a) Device statistics X

Field ID	Description
DispFrames	Video Frame Rate

(b) Service metric Y

Table 1: Device statistics X and service metric Y. Rates are given in units per second.

For this project, we say that the VoD service conforms to the SLA at a particular time, if $Y \geq 18 frames/second$ at that time; otherwise, we say that the VoD service violates the SLA.

Figure 1 is a simplification of a practical system. It does not take into account network statistics and client device statistics. Also, the service platform is generally much more complex.

Project tasks

The project is composed of three tasks.

Task I - Data Exploration

The objective of this task is to familiarize you with Python-based data exploration tools. We use the above mentioned trace with 3600 observations as the data set for the following computations.

1. Compute the following statistics for each component of X and Y: mean, maximum, minimum, 25th percentile, 90th percentile, and standard deviation.
2. Compute the following quantities:
   (a) the number of observations with memory usage larger than 80%;
   (b) the average number of used TCP sockets for observations with more than 18000 interrupts/sec;
   (c) the minimum memory utilization for observations with CPU idle time lower than 20%.
3. Produce the following plots:
   (a) Time series of percentage of idle CPU and of used memory (both in a single plot);
   (b) Density plots, histograms, and box plots of idle CPU and of used memory.

Resources:

- "Anaconda Python data science package (includes Jupyter Notebook, Python 2.7)" https://www.anaconda.com.
- "Scikit Learn" http://scikit-learn.org.
- Libraries included in Anaconda; Jupyter notebook help has pointers to descriptions:
  - pandas: Python data analysis
  - numpy: N-dimensional array package
  - matplotlib: 2D plotting package

Task II - Estimating Service Metrics from Device Statistics

The objective of this task is to estimate the frame rate of a VoD service from the device statistics of a VoD server (see Figure 1). Our approach is to gather observations from the system and apply linear regression on this data to estimate the service metric Y from device statistics X. For this task, we use the trace with 3600 observations described above. The device statistics are described by the feature set in Table 1(a), and the service metric is shown in Table 1(b).

Your task is to compute (i.e., to train) a linear model M that accurately maps device statistics onto service metrics.

You train and test your model M with the so-called validation-set technique. This technique entails that you split the set of observations into two parts: the *training set* for computing the model M and the *test set* for evaluating the accuracy of M. From the complete set of observations, you select uniformly at random 70% of the observations (i.e., 2520 observations) to form the training set and then assign the remaining 30% (i.e., 1080 observations) to the test set.

1. **Evaluate the Accuracy of Service Metric Estimation**

   (a) Model Training - use linear regression to train a model M with the training set. Provide the coefficients $(\Theta_0, ..., \Theta_9)$ of your model M. (Θ_0 is the offset.)

   (b) Accuracy of Model M - compute the *estimation error* of M over the test set. We define the estimation error as the *Normalized Mean Absolute Error* $(NMAE) = \frac{1}{\bar{y}}(\frac{1}{m}\sum_{i=1}^{m}|y_i - \hat{y}_i|)$, whereby $\hat{y}_i$ is the model estimation for the measured service metric y_i, and $\bar{y}$ is the average of the observations y_i of the test set, which is of size $m = 1080$ [1]. Note that $\hat{y}_i = M(y_i)$.
   As a baseline for M, use a naïve method which relies on Y values only. For each $x \in X$ it predicts a constant value $\bar{y}$ which is the sample mean of the samples y_i in the training set. Compute $\bar{y}$ for the naïve method for the training set and compute the NMAE for the test set.

   (c) Produce a time series plot that shows both the measurements and the model estimations for M for the Video Frame Rate values in the test set (see example of such a plot in Figure 4(a) of [1]). Show also the prediction of the a naïve method.

   (d) Produce a density plot and a histogram for the Video Frame Rate values in the test set. Set the bin size of the histogram to 1 frame.

   (e) Produce a density plot for the prediction errors $y_i - \hat{y}_i$ in the test set.

   (f) Based on the above figures and graphs, discuss the accuracy of estimating the Video Frame Rate.

2. **Study the Relationship between Estimation Accuracy and the Size of the Training Set**

   (a) From the above training set with 2520 observations, create six training sets by selecting uniformly at random 50, 100, 200, 500, 1000, and 2520 observations (which is the original set).

   (b) Train a linear model and compute the $NMAE$ for each model for the original test set with 1080 observations.

   (c) Perform the above 50 times, so you train models for 50 different subsets of a given size.

   (d) Produce a plot that shows $NMAE$ for M against the size of the training set. Use error bars or box plots to show the range of the $NMAE$ values for a given set size.

   (e) Based on the above, discuss the relationship between the accuracy of the model estimations and the training set.

Programming Resources:

- For computing the linear model, use `sklearn.linear_model.LinearRegression`.
- For splitting a set into training set and test set, use `sklearn.model_selection.train_test_split`.

- For producing a plot with error bars, you can use `matplotlib.axes.Axes.errorbar`.
- For producing a density plot, histogram or boxplot, you can use `pandas.DataFrame.plot`.

Machine Learning Resources:
If you are not familiar with the basic concepts of machine learning or the method of linear regression, we recommend a list of short videos by Andrew Ng from Stanford University. You can find them at:
`https://www.youtube.com/playlist?list=PLLssT5z_DsK-h9vYZkQkYNWcItqhlRJLN`
Specifically, watch the videos 1.1, 1,2, 2.1, 2.2, 2.3, 2.4, 4.1 .

Task III - Estimating SLA Conformance and Violation from Device Statistics

The objective for this task is to build a binary classifier function that estimates whether the VoD service conforms to the given SLA (see above) for specific device statistics X, or whether the service violates the SLA for a specific value of X.

You apply logistic regression to build the classifier. As in the previous task, build a training set and a test set from the trace data, containing 70% of the observations and 30% of the observations, respectively.

1. Model Training - use Logistic Regression to train a classifier C with the training set. Provide the coefficients $(\Theta_0, ..., \Theta_9)$ of your model C. (Θ_0 is the offset.)
2. Accuracy of the Classifiers C - Compute the classification error (ERR) on the test set for C. For this, you first compute the confusion matrix, which includes the four numbers True Positives (TP), True Negatives (TN), False Positives (FN), and False Negatives (FN). We define the classification error as $ERR = 1 - \frac{TP+TN}{m}$, whereby m is the number of observations in the test set. A true positive is an observation that is correctly classified by the classifier as conforming to the SLA; a true negative is an observation that is correctly classified by the classifier as violating the SLA.
3. As a baseline for C, use a naïve method which relies on Y values only, as follows. For each $x \in X$, the naïve classifier predicts a value $True$ with probability p and $False$ with probability $1-p$. p is the fraction of Y values that conform with the SLA. Compute p over the training set and the classification error for the naïve classifier over the test set.
4. Build a new classifier by extending extend the linear regression function developed in Task II with a check on the output, i.e., the Video Frame Rate. If the frame rate for a given X is above the SLA threshold, then the Y label of the classifier is set to conformance, otherwise to violation. Compute the new classifier over the training set and the classification error for this new classifier over the test set.
5. Formulate your observations and conclusions based on the above work.

Programming Resources:

- For logistic regression, use `sklearn.linear_model.LogisticRegression` (liblinear solver).

Machine Learning Resources:
You can learn about logistic regression at:
`https://www.youtube.com/playlist?list=PLLssT5z_DsK-h9vYZkQkYNWcItqhlRJLN`
Specifically, watch the videos 6.1, 6,2, 6.3, 6.4.

Task IV - Reduce the Number of Device Statistics to Estimate the Service Metric

The objective for this task is to reduce the number of device statistics X needed for accurately estimating Y. Since set of size n has 2^n subsets, finding a minimal of X that allows to predict Y with the smallest error for a specific learning method is generally not feasible.

You will use two heuristic methods two achieve this task.

1. Construct a training set and a test set from the trace as above.
2. Method 1: Build all subsets of the feature set X that contain either one or two features (i.e., device statistics). Compute the models for each of these sets for linear regression over the training set. Plot a histogram of the error values (NMAE) of all the models for the test set. Identify the feature set that produces the model with the smallest error and give the device statistic(s) in this set.
3. Method 2: Linear univariate feature selection. Take each feature of X and compute the sample correlation of the feature with the corresponding Y value over the training set. For observations x_i, y_i, the sample correlation is computed as $\frac{1}{m}\sum_{i=1}^{m}(x_i - \bar{x})(y_i - \bar{y})/(\sigma_X * \sigma_Y)$ whereby $\bar{x}$ and $\bar{y}$ are sample means and m is the size of the training set; σ_X is the standard deviation $\sqrt{(\frac{1}{m}\sum_{i=1}^{m}(x_i - \bar{x})^2)}$ and likewise for σ_Y. The correlation values fall into the interval $[-1, +1]$. Rank the features according to the square of the correlation values; the top feature has the highest value. Build nine feature sets composed of the top k features, $k = 1..9$. Compute the model for each of these nine sets for linear regression over the training set and compute the error (NMAE) of these models over the test set. Produce a plot that shows the error value in function of the set k.
4. Describe your observations and conclusions.

Optional Task

Since the feature set X contains only nine features, it is feasible in this case to compute and evaluate models for all possible subsets of the set X and find the subset with the smallest error on the test set.

1. Compute models and for all subsets of X and evaluate them by computing their errors (NMAE).
2. Plot a histogram of the model errors. Set the bin size of the histogram to 0.01.
3. For the subsets of size 1 to 9, identify the features (i.e., device statistics) that give the smallest error.
4. Produce a plot that shows NMAE versus subset size 1 to 9. For each subset, show an error bar or a boxplot with the range of NMAE values.
5. Describe your observations and conclusions.

Deliverables and Grading

At each deadline (see web page for the dates), you submit two files, a report in pdf and a Jupyter notebook file with the code.

The reports include all output values, tables, plots, discussions for the particular task(s). For deadline 1, submit a report for Task I and II; for deadline 2, submit a report for Task III; for deadline 3, submit a complete report for all tasks. The same applies for the Jupyter notebook file.

Label the plots and the tables and refer to the labels in the text. Proofread your report before submission.

Name your files $Your\_name.Task(s)$*.pdf* or $Your\_name.Task(s)$*.ipynb*, respectively.

In Jupyter notebook, read in the data as follows:

```
import pandas as pd
#read external data into panda data frames
X = pd.read_csv('data/X.csv')
Y = pd.read_csv('data/Y.csv')
```

Provide comments in your code like in the above example.

Project Interview

After submitting the last report, you step by for an interview of about 15 minutes. Be prepared to answer questions related to the project and your report. The schedule will be made available via Canvas.

Grading is performed according to Table 2.

	Deadline 1	Deadline 2	Deadline 3	
Task I,II	2	1	0	0
Task III	–	2	1	0
All Tasks	–	–	2	0
Interview	–	–	–	4

Table 2: Grading of deliverables and interview.

References

[1] R. Yanggratoke, J. Ahmed, J. Ardelius, C. Flinta, A. Johnsson, D. Gillblad, and R. Stadler, "Predicting real-time service-level metrics from device statistics," tech. rep., 2014. `http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-152637`.

[2] R. Yanggratoke, J. Ahmed, J. Ardelius, C. Flinta, A. Johnsson, D. Gillblad, and R. Stadler, "Linux kernel statistics from a video server and service metrics from a video client.," 2014. Distributed by Machine learning data set repository [MLData.org]. `http://mldata.org/repository/data/viewslug/realm-im2015-vod-traces`.