EP2420/EP272V Project 1
Estimating Service Metrics from Device Measurements

Rolf Stadler Forough Shahab Xiaoxuan Wang

October 18, 2023

Project Objective and Approach

The objective of this project is to investigate the service quality 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 metric Y refers to statistics on the client side. Table 1 describes the metrics X and Y used in the project.

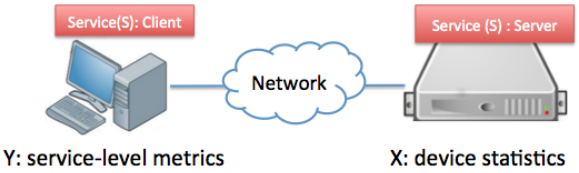

Figure 1: System configuration for estimating service metrics.

Feature	Description
runq-sz	Run queue length
%%memused	Percentage of used memory
proc/s	Rate of process creation
cswch/s	Rate of context switching
all_%%usr	Percentage of CPU utilization
ldavg-1	Load average for the last minute
totsck	Number of used sockets
pgfree/s	Rate of freeing pages
plist-sz	Number of tasks in the task list
file-nr	Number of file handles
idel/s	Number of IP datagrams delivered to IP user
tps	Rate of I/O requests to physical devices

(a) Device statistics X. See details in [2]

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.

Using machine-learning techniques, the problem is to find a function (i.e., to train a model) $M : X \to \hat{Y}$, such that $\hat{Y}$ closely approximates Y for a given X. If Y is a numeric value, the problem is referred to as

a regression problem. (If Y is a class label, the problem is called a classification problem.) To train M, measurement pairs (or observations) of the form (x_t,y_t) are needed. A set of measurement pairs, indexed by time, is called a trace. We use in this project a trace of 3600 observations, collected once every second from the system over the course of an hour. It is described in [1].

The setup depicted in Figure 1 simplifies a real system. It does not take into account network statistics and client device statistics. Also, the service platform of a commercial system is generally much more complex.

Project tasks

The project includes six tasks and Task 1 to Task 5 are mandatory. At each deadline shown below, you submit two files, a report in pdf and a Jupyter notebook file with the code. Name your files Your_name.Task(s).pdf or Your_name.Task(s).ipynb, respectively.

1. Week 1: Task 1 and Task 2.1.
2. Week 2: Task 2.2 and Task 3.1.
3. Week 3: Task 3.2 and Task 4.
4. Week 4: Task 5 and final report.
5. The optional task can be submitted any week.

Task 1 - 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 feature of X and target of Y: mean, standard deviation, maximum, minimum, 25th percentile, 50th percentile, and 90th percentile. Give three significant digits, for instance, 52.3 or 5.23e+01.
2. Compute the following statistics:
   (a) The number of observations with CPU utilization ("all_%%usr") lower than 50% and memory utilization ("%%memused") lower than 25%;
   (b) The average number of used sockets ("totsck") for observations with less than 50 000 context switches per seconds ("cswch/s").
3. Produce the following plots and describe them briefly:
   (a) Time series plots of memory usage ("%%memused") and of CPU utilization ("all_%%usr"), both curves in a single figure.
   (b) Box plots of memory usage ("%%memused") and of CPU utilization ("all_%%usr") in a single figure.
   (c) Density plots of memory usage ("%%memused") and of CPU utilization ("all_%%usr") in two figures.
   (d) Histograms of memory usage ("%%memused") with bin size 1% and of CPU utilization ("all_%%usr") with bin size 5% in two figures.

Resources:

- "Anaconda Python data science package (includes Jupyter Notebook, Python 3 or higher)", 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 package
  - numpy: N-dimensional array package
  - matplotlib: 2D plotting package
  - sklearn.preprocessing: Data preprocesssing package

Read the description of Task 2 and answer the question:

- Assume we want to predict a certain service metric from network measurements using linear regression. Under which condition do we expect the linear regression model to give predictions with high accuracy, and under which condition do we expect low prediction accuracy?

Task 2 - Estimating Service Metrics from Device Statistics using Linear Regression

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 (x_t, y_t) from the system and apply linear regression to this data in order to estimate the service metric Y from device statistics X. For this task, we use the trace with 3600 observations mentioned 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 at every point in time.

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 on unseen data. 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.

2.1 Evaluate the Accuracy of Service Metric Estimation

1. **Model Training** - use linear regression to train a model M with the training set. Provide the coefficients $(\Theta_0, ..., \Theta_{12})$ of your model M. (Θ_0 is the offset.) For the value of a coefficient, give three significant digits, for example, -0.0532 or -5.32e-2.

2. **Accuracy of Model** M - compute the *estimation error* of M on 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 $m = 1080$. Note that $\hat{y}_i = M(y_i)$. Give the NMAE of Model M for the test set.
   As a baseline for M, use a naïve method which relies on Y values only. For each $x \in X$ it predicts the constant value $\bar{y}$ which is the mean of the samples y_i in the training set. Compute $\bar{y}$ for the naïve method on the training set and give the NMAE of naïve method for the test set.

3. 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. Show also the prediction of the a naïve method (see Figure 2). Sort the samples according to the timestamp for the time series plot.

4. 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/second.

5. Produce a density plot of the estimation errors $y_i - \hat{y}_i$ on the test set.

6. Based on the above figures and graphs, discuss the accuracy of estimating the Video Frame Rate and compare the accuracy of linear regression with that of naïve method.

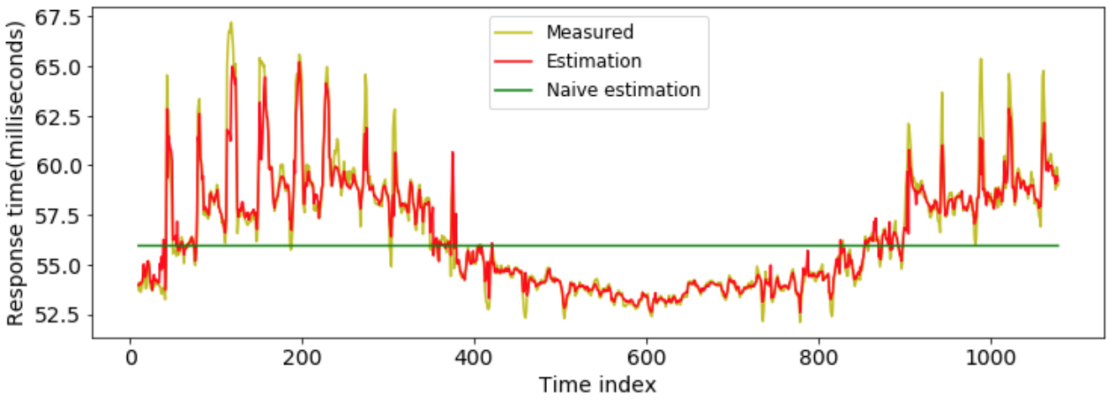

Figure 2: Example time series plot: Key-Value store service

2.2 Study the Relationship between Estimation Error and the Size of the Training Set

1. From the above trace of observations S with 3600 observations, create a test set T by selecting uniformly at random 1000 samples. Then, create six training sets $S_1, ..., S_6$ by selecting uniformly at random 50, 100, 200, 400, 800, and 1600 observations from S which have not been chosen for T.
2. Train a linear model for each training set S_i and compute the NMAE for this model on the test set T.
3. Perform the above 50 times, so you train and evaluate models for 50 different pairs of training set and test set for a given training set size.
4. Produce a plot that shows NMAE for M (vertical axis) against the size of the training set (horizontal axis). Use error bars to indicate the standard deviation or box plots to show the range of the NMAE values for a given set size.
5. Based on the above, discuss the relationship between the estimation error and the size of the training set.

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.
- For computing the linear model, use `sklearn.linear_model.LinearRegression`.
- For splitting a set into the training set and test set, use `sklearn.model_selection.train_test_split`.
- For producing a plot with bars of the variance, you can use `matplotlib.axes.Axes.errorbar`.
- For producing a density plot, histogram or boxplot, you can use `pandas.DataFrame.plot`.

Read the description of Task 3 and answer the question:

- What is the computational complexity of the optimal method and the heuristic method in Task 3 in function of the size of the original feature set (which is the same as the dimensionality of the feature space)?

Task 3 - Reduce the Number of Device Statistics to Estimate the Service Metric

The objective of this task is to reduce the number of device statistics (features) of X needed for accurately estimating Y. Since a feature set of size n has 2^n subsets, finding a minimal subset of the statistics of X that predicts Y with the smallest error is not feasible for large n. The data collection and training overhead would be too large.

You will investigate two methods for feature selection. Construct a training set and a test set from the trace as in Task 2.1.

3.1 Optimal Method

1. This method is appropriate if X contains a small number of features: Build all subsets of the feature set X. Using the training set, compute a linear regression model for each of these feature subsets. For each model compute the error (NMAE) on the test set. Provide the features of the model with the smallest error. Produce a figure that contains 12 box plots, one box plot each with the errors (NMAE) for all models that contain 1 feature, 2 features, ..., 12 features.

2. Record the computing time of the optimal method (i.e. the time for training and for evaluating for all possible subsets).

3.2 Heuristic Method

1. This method uses a univariate feature selection technique. Take each feature of X and compute the Pearson correlation of the feature with the Y value on the training set. For feature j and target y, the Pearson correlation is computed as $r_{x_{:,j},y} = \frac{1}{m}\sum_{i=1}^{m}(x_{i,j} - \bar{x_{:,j}})(y_i - \bar{y})/(\sigma_{x_{:,j}} * \sigma_y)$ whereby $\bar{x_{:,j}}$ and $\bar{y}$ are sample means and m is the size of the training set; $\sigma_{x_{:,j}}$ is the standard deviation of the $x_{:,j}$ values, namely, $\sigma_{x_{:,j}} = \sqrt{(\frac{1}{m}\sum_{i=1}^{m}(x_{i,j} - \bar{x_{:,j}})^2)}$, 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 twelve feature sets composed of the top k features, $k = 1, ..., 12$. The first feature set contains the top feature, the second feature set contains the top two features, etc. For each feature set, compute the linear regression model on the training set and compute the error (NMAE) on the test set. Produce a plot that shows the error value in function of the set size k.

2. Record the computing time of the heuristic method (i.e. the time for computing the Pearson correlations between all pairs of a feature column and the target column).

A correlation matrix is a square matrix whose cells show the Pearson correlation between column vectors. The columns represent the features and the target, i.e. $x_{:,1}, ..., x_{:,n}, y$. Plot the heat map of this correlation matrix. Analyze and explain the results of the optimal method and the heuristic method based on the correlation between the features and the video frame rate.

Compare the optimal method with the heuristic method as follows. Produce a single plot with two curves. The vertical axis is NMAE and the horizontal axis is the size k of the feature set. The first curve shows the minimum error of the optimal method for feature sets of size k. The second curve shows the error of the heuristic method for the sets with the top k features. Compare the two curves and explain the difference in error values. What can you conclude from this graph and from the other results of this task?

Resources:

- For producing a heat map, you can use `seaborn.heatmap`.
- For computing the Pearson correlation, you can use `numpy.corrcoef`.

Read the description of Task 4 and answer the question:

- We compare three dimensionality reduction methods: the optimal method, the heuristic method, and PCA. What are the advantages and disadvantages of using PCA (see Task 4) compared to using the optimal method or the heuristic method?

Task 4 - Dimensionality reduction using Principal Component Analysis (PCA)

PCA is a traditional method for linear dimensionality reduction using Singular Value Decomposition of the data to project it to a lower-dimensional space.

1. Split the device statistics X into a training set and a test set. Use PCA on the training set to reduce the dimensionality of the feature space to $k = 1, 2, ..., 12$. This produces 12 linear mappings $\mathbb{R}^{12} \rightarrow \mathbb{R}^k$. For each k map the original training set into a training set in the reduced space. Do the same with the original test set. For each k train a linear regression model using the mapped training set. Evaluate the error (NMAE) of this regression model using the mapped test set. Finally, produce a plot that shows the error value in function of the dimensionality k of the reduced feature space.

2. Produce a plot with three curves to compare the performance of the two feature reduction methods studied in Task 3 and Task 4 for dimensionality reduction. This plot includes the two curves constructed in Task 3 and adds the third curve describing PCA. Compare the three methods based on this plot.

3. Record the computing time of PCA (time of fitting and transforming from the original space to the reduced space).

4. Compare the three dimensionality reduction methods regarding computational overhead and prediction accuracy. Describe scenarios for which each of these methods would be preferable.

Resources:

- You can learn about PCA at:
  `https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html`.

Read the description of Task 5 and answer the question:

- Which main strategies can be used for hyper-parameter search in Task 5? Choose one of them and describe it briefly.

Task 5 - Estimating Service Metrics from Device Statistics using Neural Network Regression

The objective of this task is to estimate a service metric Y from device statistics X. In contrast to Task 2, you use a neural network regressor, which is a non-linear function, instead of a linear regressor.

1. **Model Training** - configure a neural network and train a model M with the training set. Use the Keras library for this task.

2. **Hyper-parameters Search** - identify effective hyper-parameters for the optimizer, for the regularizer, and for the architecture configuration:

   (a) Optimizer: The learning rate and the number of epochs;

   (b) Regularizer: The type of norm (l^1 or l^2) and the value of lambda (λ);

   (c) Architecture: the number of hidden layers and the number of nodes in each hidden layer.

   Describe how you conduct the hyper-parameter search.

3. **Accuracy of Model** M - compute the *estimation error* (NMAE) of M on the test set.
   As a baseline for M, use the naïve method described in Task 2.

4. Produce a time series plot that shows both the measurements of and the model estimations for the Video Frame Rate values in the test set. Show also the predictions of the naïve method.

5. Produce a density plot of the estimation errors $y_i - \hat{y}_i$ on the test set.

6. Based on the above evaluation, discuss the accuracy of estimating the Video Frame Rate. Discuss the role of hyper-parameter search for obtaining an effective model (you may compare the prediction error before and after hyper-parameter tuning). Compare the results with linear regression in Task 2. (If the hyper-parameters are tuned, neural network regression usually outperforms linear regression.)

Resources:

- You can learn about Keras, hyper-parameter search, and regularization at:
  `https://keras.io/`
  `https://www.youtube.com/watch?v=u73PU6Qwl1I`
  `https://www.coursera.org/lecture/deep-neural-network/regularization-Srsrc`
  `https://www.coursera.org/lecture/deep-neural-network/tuning-process-dknSn`

Optional Task - Estimating Service-Level Agreement (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 a given SLA for specific device statistics X, or whether the service violates the SLA for a specific value of X. 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.

You apply logistic regression to build the classifier. As in Task 2, build a training set containing 70% and a test set containing 30% of the observations.

1. **Model Training** - use Logistic Regression to train a classifier C with the training set. Provide the coefficients $(\Theta_0, ..., \Theta_{12})$ of your model C. (Θ_0 is the offset.) For the value of a coefficient, give three significant digits, for example, -0.0532 or -5.32e-2.

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 as conforming to the SLA; a true negative is an observation that is correctly classified as violating the SLA. Plot the confusion matrix (see Figure 3). Compare the values of FP and FN. What can you conclude from this comparison?

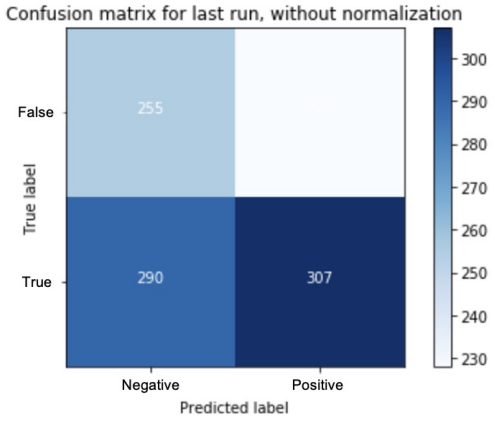

Figure 3: Example confusion matrix

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 on the training set and the classification error for the naïve classifier on the test set.

4. Build a new classifier by extending the linear regression function developed in Task 2 with a check on the output, i.e., the Video Frame Rate. If the estimated 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 on the training set and compute the classification error for this new classifier on the test set.

5. Compare the above three classifiers regarding the classification error (ERR). Can you explain why the errors differ?

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.
- For logistic regression, use `sklearn.linear_model.LogisticRegression`.

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," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pp. 414–422, IEEE, 2015.

[2] sar(1), "Linux manual page." [Online]. Available at: `https://man7.org/linux/man-pages/man1/sar.1.html`, Accessed on: October 5, 2023.