EP2420 *Project 1*
Estimating Service Metrics from Device Measurements

Cheng Yang

November 21, 2020

Project Overview

The scenario of the project is shown in Figure 1. The client machine is connected to a server via a network and accesses the video-on-demand (VoD) service that runs on the server. In this setting, device statistics X refer to operating-system metrics on the server side, while service metric Y(Video Frame Rate) refer to statistics on the client side. The interpretations of the features of X are shown in Table 1.

Our target is to use the device measurements X to predict the service metrics Y using linear regression and neural network and evaluate the models from various kinds of approaches.

The project is performed using Python and the related packages are Numpy, SciPy, Matplotlib, Scikit-learn, Pandas, Tensorflow, Keras.

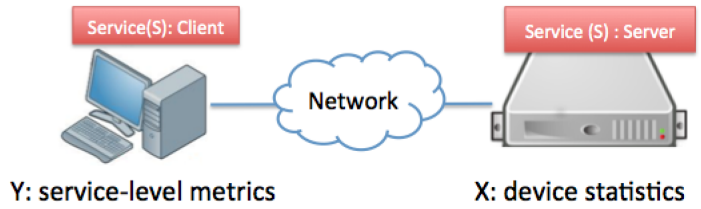


Figure 1: System configuration for estimating service metrics.

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 r	Number of IP datagrames delivered to IP use
tps	Rate of transfers to physical devices

Table 1: Device statistics X

Data Sets and Data Pre-processing

There are 3600 observations indexed by the captured time in the data set with 12 different features, collected once every second from the system over the course of an hour.

We will explore the data in Task I. As for the data pre-processing, we only apply standardization to the device measurements X and it will be used in Task IV and Task V.

Task I - Data Exploration

In this task, we will perform some basic data exploration to get some knowledge of the data set, which would be quite useful for the later prediction.

1. Compute the following statistics for each feature of X and target of Y: mean, standard deviation, Maximum, minimum, 25th percentile, and 90th percentile.

	runq-sz	%%memused	proc/s	cswch/s	all_%%usr	ldavg-1	totsck	\
count	3600.00	3600.00	3600.00	3600.00	3600.00	3600.00	3600.00	
mean	52.67	26.22	6.18	43121.73	73.07	60.39	455.19	
std	47.37	6.25	9.06	24722.83	27.70	50.44	181.43	
min	3.00	15.01	0.00	8380.00	17.51	4.60	269.00	
25%	11.00	22.91	0.00	18761.25	45.03	13.39	311.00	
50%	33.00	25.80	0.00	38849.50	93.47	43.99	366.00	
90%	126.00	35.91	19.00	73280.60	98.12	138.59	753.00	
max	199.00	38.99	50.00	85020.00	98.62	186.86	958.00	

	pgfree/s	plist-sz	file-nr	idel/s	tps	DispFrames
count	3600.00	3600.00	3600.00	3600.00	3600.00	3600.00
mean	73850.67	802.12	2640.43	45.60	6.25	20.28
std	28695.80	346.23	196.92	256.02	12.70	4.99
min	16726.00	452.00	2352.00	1.00	0.00	0.00
25%	56051.25	525.00	2496.00	11.00	0.00	14.77
50%	71476.50	632.00	2592.00	21.00	0.00	23.41
90%	112971.40	1370.10	2928.00	61.00	18.00	25.00
max	442723.00	1772.00	3312.00	9013.00	90.00	30.36

Figure 2: Basic feature statistics

We can use the *pandas.describe* function to perform this task and the results are shown in Figure 2. What needs to be pointed out is that the standard deviation is the unbiased one. Firstly, the counts of all the features are 3600 and it means that there is not any missing data. Then we check the the minimum value and maximum value to see if there are any abnormal data. The maximum value of the features look reasonable(note that the maximum value of "%%*memused*" and "*all*%%*usr*" are less than 100%). The minimum value of $'tps'$ is zero which means no transfers to physical devices and except for that all the data are positive value. So far all the data looks reasonable.

2. Compute the following statistics:

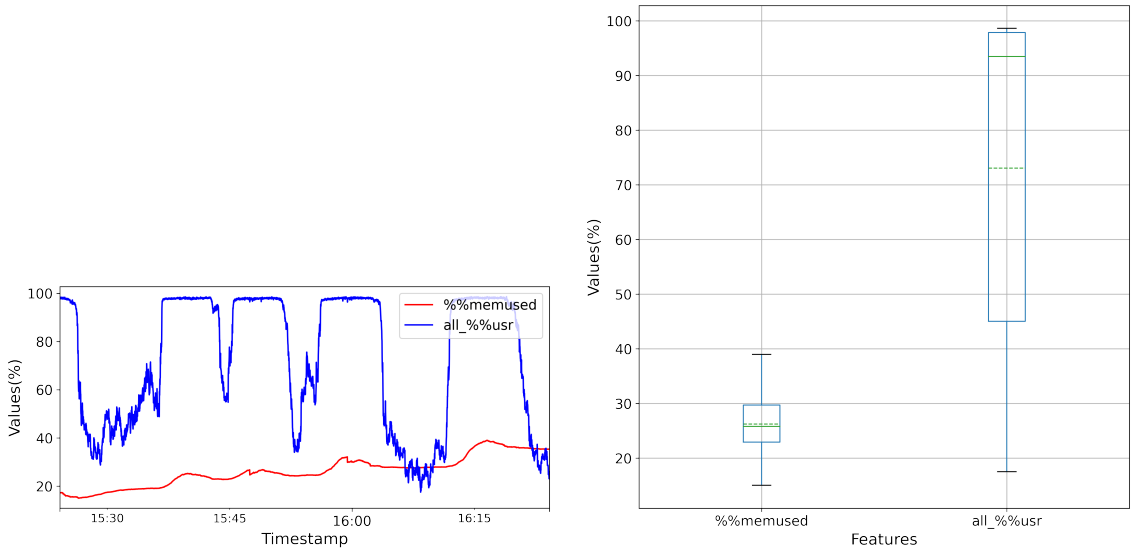
   (a) The number of observations with CPU utilization smaller than 90% and memory utilization smaller than 50% is 1746. Since the maximum value of the memory utilization is 38.99% which is smaller than 50%, it means that about half of the time CPU utilization exceeds 90%.

   (b) The average number of used sockets for observations with less than 60000 context switches per seconds is 326.14.

3. Produce the following plots:

   (a) Time series plot of memory usage ("%%*memused*") and CPU utilization ("*all*%%*usr*"), both curves in a single plot.

   The result is shown in Figure 3(a). The CPU utilization rate is close to 100% for several durations and has certain periodic characteristics. The memory utilization rate is slowly increasing over time, and its correlation with the CPU utilization rate is not very obvious.



(a) Time series plot of memory usage and CPU utilization (b) Box plot of memory usage and CPU utilization

Figure 3: Plots for 1.3(a) and 1.3(b)

(b) Box plot of memory usage ("%%*memused*") and CPU utilization ("*all_%%usr*") in a single plot. From the result shown in Figure 3(b), we learn that the average level and variation of CPU utilization are much higher than memory utilization.

(c) Density plots of memory usage ("%%*memused*") and CPU utilization ("*all_%%usr*") in two plots.

(d) A histograms with bin size 5% of memory usage ("%%*memused*") and a histogram with bin size 1% of CPU utilization ("*all_%%usr*") in two plots.

The results are shown in Figure 4. The density plots verifies the conclusion we got from the box plot. And what is interesting is that the density distribution of memory utilization has four peaks, at around 18%, 24%, 28%, and 36%, while the density distribution of CPU utilization has two peaks, at around 35% and 98%.

Histogram plot is the discrete version of density plot whose advantage is that we can directly see the number of statistics(frequency). The bin size of 5% is too small for the data set and does not reflect the four peaks shown in the density plot clearly, so we change the bin size to 0.5%.

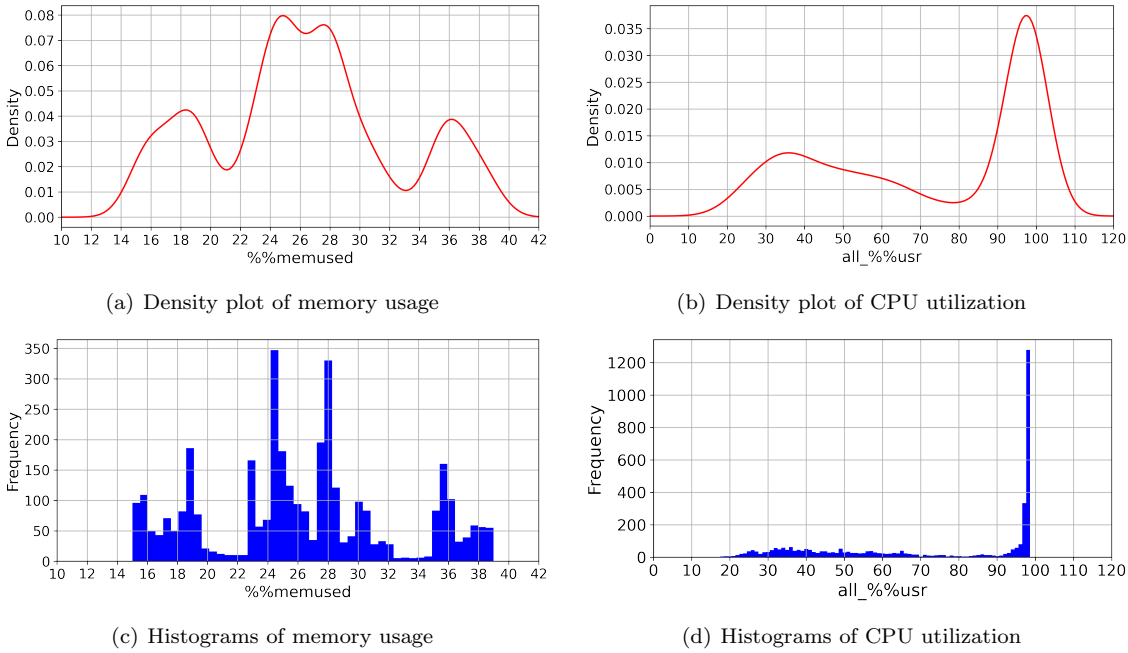
Task 2 - Estimating Service Metrics from Device Statistics using Linear Regression

The objective of this task is to estimate the frame rate from the device statistics by training a linear regression model. The training and evaluation of the model makes use the Python package Scikit-learn.

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 coeffcients $(\Theta_0..., \Theta_{12})$ of your model M. (Θ_0 is the offset.)

   The linear regression coefficients of the features sorted in Table 1 are : [-2.99e-02 -7.47e-02 6.15e-03 -1.96e-04 1.32e-01 -1.36e-02 7.04e-03 -1.79e-05 -6.18e-03 2.90e-03 -1.92e-04 1.28e-03] and the intercept is 18.8.



(a) Density plot of memory usage

(b) Density plot of CPU utilization

(c) Histograms of memory usage

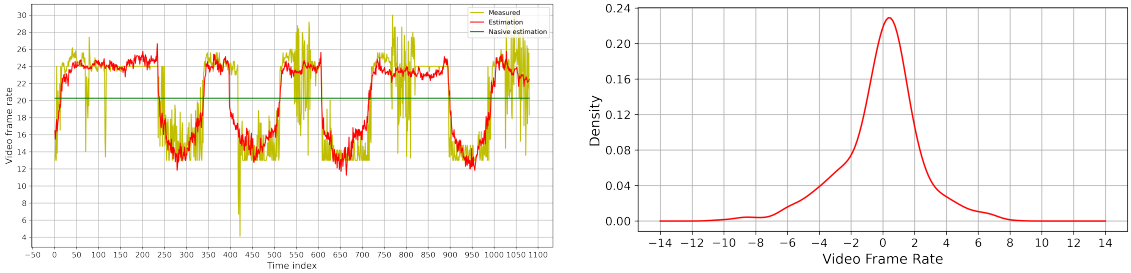
(d) Histograms of CPU utilization

Figure 4: Density plots and histograms of memory usage and CPU utilization

2. Accuracy of Model M - compute the estimation error of M on the test set. We denote the estimation error as the *Normalized Mean Absolute Error* ($NMAE$).
   As a baseline for M, use a naive method that predicts the constant value $\overline{y}$ which is the mean of the samples $y_i$in the training set. Compute the NMAE of the linear model and the naive method.

   NMAE of the linear model and the naive estimation are 0.092 and 0.22.

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 naive method.



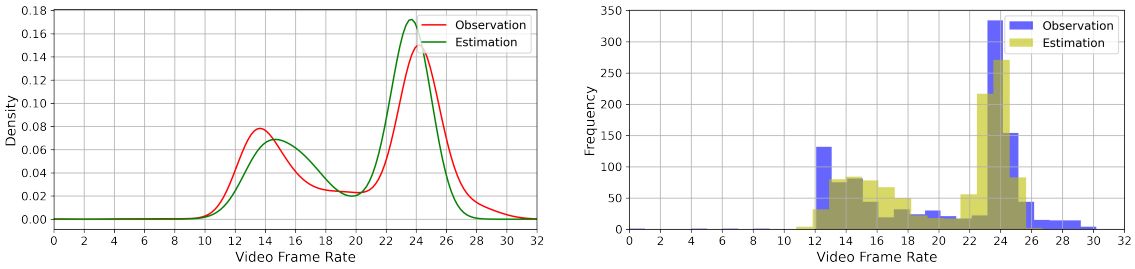
(a) Measurements, linear regression estimations and naive estimation for the Video Frame Rate values

(b) Density plots of the Video Frame Rate error in the test set

Figure 5: Plots for 2.1.3 and 2.1.5

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.

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



(a) Density plot

(b) Histograms

Figure 6: Density plots and histograms of the Video Frame Rate values in the test set

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

   The plots of 2.1.3 to 2.1.5 are shown in Figure 5. From the time series plot, we can intuitively see that the linear model follows the changes in the test set data at a macro level, but it does not fit well in the fluctuation of the details.

   From Figure 6, it turns out that our linear model have successfully modeled the Bimodal distribution of the test data.

   It can be seen from the density plot that the probability distribution of the error is roughly symmetric about 0, and our linear modal looks reasonable so far.

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 six training sets $S_1, \ldots, S_6$ by selecting uniformly at random 50, 100, 200, 400, 800, and 1600 observations. For each training set S_i, you create a test set T_i, by selecting uniformly at random 1000 samples from S which have not been chosen for Si. You end up with six test sets $T_1, \ldots, T_6$.

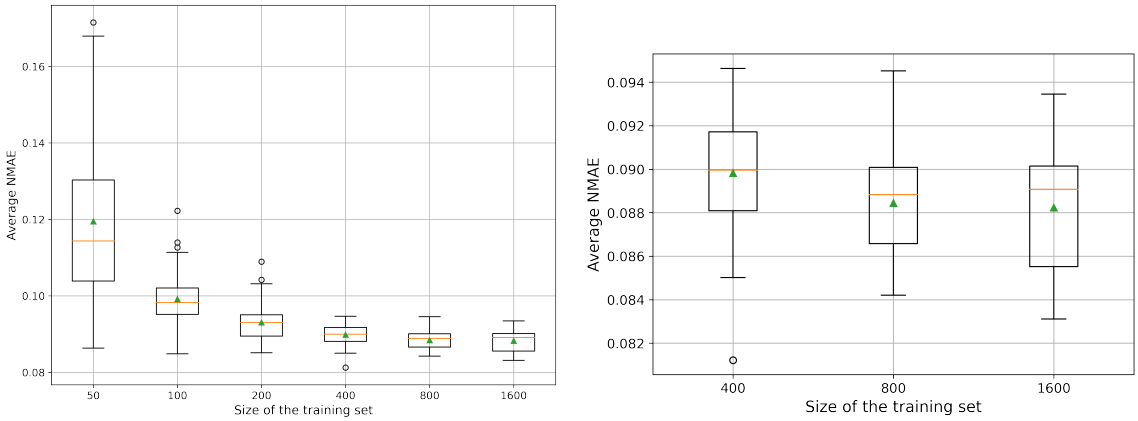
2. Train a linear model for each training set Si and compute the NMAE for this model on the corresponding test set $T_i, i = 1, \ldots, 6$.

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 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.

   Firstly, we need to theoretically study the relationship between estimation error and the size of the training set. There are two kinds of errors here. The first is caused by the difference between the linear model and the real model, which means that we have ignored some factors or considered some irrelevant factors in the model. The second type of error is caused by the noise of the data itself. As the size of the training set increases, this error can be gradually reduced. Therefore, as the size of the training set increases, the total error level will decrease and approach the first type of error level.
   It can be seen from Figure 7(a) that the average error decreases with the increase in the number of training sets, and the decline gradually slows down. The average error levels of the training set sizes of 800 and 1600 are almost the same. This verifies our theoretical analysis.
   Besides, we need to consider the impact of randomly selecting the training set and test set on the test results. To reduce this impact, we conducted 50 tests for every size of the training set. But this kind of



(a) (b)

Figure 7: Box plot of 50 times average NMAE of linear regression with training set size of 50, 100, 200, 400, 800, 1600 and test set size 1000

influence still exists. The most direct proof is that the minimum error is not obtained in the experiment with the largest training set, but is obtained when the training set size is 400 points. Similarly, we can use this to explain why the median error of the training set size of 1600 is slightly greater than that of the training set size of 800 which is shown in Figure 7(b), and the level of the minimum error in 50 trials is almost independent of the training set size.
Also, we can see that as the size of the training set increases, the interquartile range(IQR) will gradually decrease (but it will still be affected by the random data set selection mentioned before), which means that the increase in the size of the training set will make the range of error fluctuations smaller, in other words, the result is more stable.

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

The objective of this task is to reduce the number of device statistics of X needed for accurately estimating Y .

1. Construct a training set and a test set from the trace as in Task 2.
   All the experiments in Part 3 are perform on the same randomly selected training set and testing set with a proportion of $7:3$.

2. Method 1 (Optimal method): 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 plot that contains 12 box plots, each with the errors (NMAE) for all models that contain 1, 2, ..., 12 features.

   The result is later shown in Figure 10(a) together with other methods. The model with the smallest error which is 0.0884 is trained with 9 features: $'runq-sz'$, $'\%\%memused'$, $'proc/s'$, $'cswch/s'$, $'all\_\%\%usr'$, $'pgfree/s'$, $'plist-sz'$, $'file-nr'$, $'idel/s'$. When the number of features is larger than 2, the minimum error within those models is on the same level(around 0.089). This means that three important features can determine the accuracy of the prediction. After inspection, we found that those models all contain three key features which are '$runq-sz'$, '$cswch/s'$, and '$all\_\%\%usr'$. And we believe that these 3 features and those ones that are high relevant to these 3 are helpful to the linear model. When the number of features is 3, 4, 5, 6, a large number of outliers appear in the box plot. This is because there are a large number of feature combinations that do not include key features but include

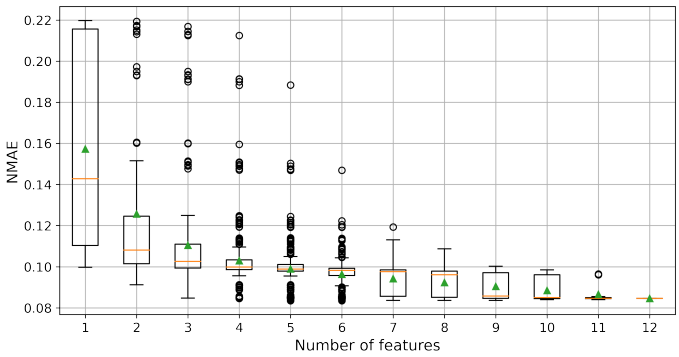


Figure 8: Box plots of linear regression NMAE with different number of features

some features that weakens the linear model. When the number of features is greater than 6, the proportion of feature combinations that do not include key features is small, which reduces the number of outliers.

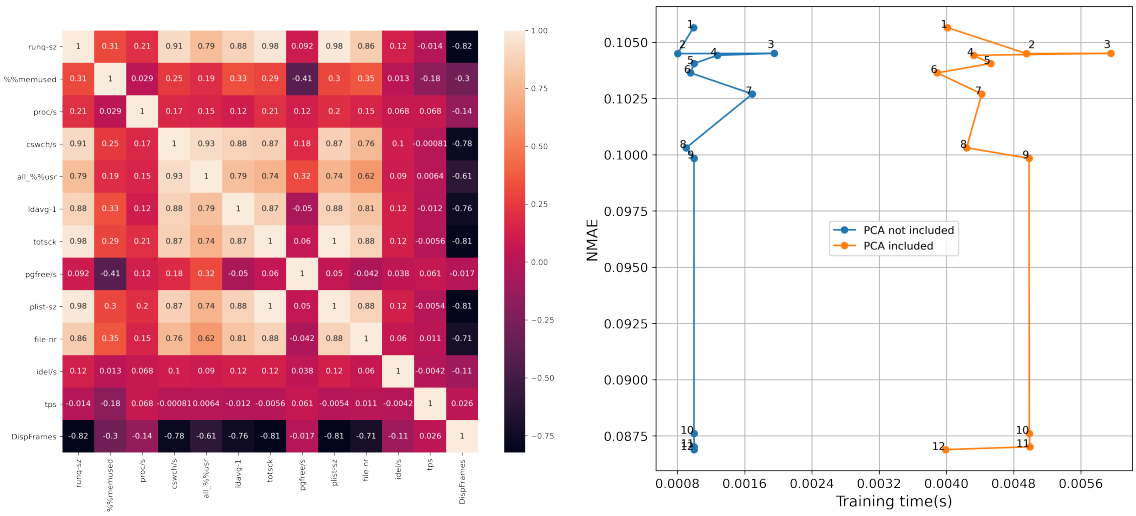
3. Method 2 (Heuristic method): Linear univariate feature selection. Take each feature of X and compute the Pearson correlation of the feature with the Y value on the training set. Rank the features according to the square of the correlation values. Build twelve feature sets composed of the top k features, $k = 1, \ldots, 12$. 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.

   The rank of the features according to the square of the correlation values from the maximum to the minimum is: $'runq-sz'$, $'plist-sz'$, $'totsck'$, $'cswch/s'$, $'ldavg-1'$, $'file-nr'$, $'all\_\%\%usr'$, $'\%\%memused'$, $'proc/s'$, $'idel/s'$, $'tps'$, $'pgfree/s'$.
   We have mentioned before that the 3 important features are '$runq-sz'$, '$cswch/s'$, and '$all\_\%\%usr'$, which are not the exactly top 3 features shown above. It means that the Pearson correlation is a very important reference, but we cannot rely on it completely. Further analysis will be included in section 3.4.
   The plot is shown in Figure 10(a). We can observed that the when the number of features changes from 6 to 7, the error has dropped significantly because of the introduction of feature $'all\_\%\%usr'$.

4. Plot the heat map of the correlation matrix. Analyze and explain the results of part 2 and 3 based on the correlation among features and video frame rate.

   The heat map is shown in Figure 9(a). The key is that we should not only consider the correlation between features and video frame rate but also consider the correlation among features. $'runq-sz'$, $'plist-sz'$, and $'totsck'$ are 3 features that are most relevant with video frame rate, but their correlation between each other are very high(0.98 or 1), which implies that $'plist-sz'$ and $'plist-sz'$ are highly linearly related to $'runq-sz'$, and we only need to use $'runq-sz'$ to represent these three features.
   And for those features that have relatively low Pearson correlation with video frame rate, they actually may contain information about video frame rate(eg. $'all\_\%\%usr'$), but the relationships are not necessarily linear, which means Pearson correlation cannot be used as the only criterion for measuring correlation.

5. Compare the optimal method with the heuristic method as follows. Produce a single plot with two curves. Compare the two curves and explain the difference in error values. What can you conclude from this graph and from the other results in this task?



(a) Heat map of the correlation matrix of the training set (b) Plot of linear regression NMAE with the training time

Figure 9: Plots of 3.4 and 4.2

The plot is shown in Figure 10(a). When the number of features is 12, the model is the same which brings the same error. When the number of features is 1, the heuristic method uses feature $'runq-sz'$ that has the highest correlation with video frame rate which works best if we only take 1 feature in the linear model. For the rest of the cases, the optimal method always works better than the heuristic method.
This is due to the correlation between features and the fact that Pearson correlation cannot be used as the only criterion for measuring correlation, whcich has been explained in section 3.4.

Dimensionality reduction using Principal Component Analysis (PCA)

The objective of this task is to use PCA to reduce the dimensionality of the input features.

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 $k = 1, 2, \ldots, 12$. For each k map the original training set into a training set in the reduced space and train a linear regression model using the mapped training set. Evaluate the error 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.

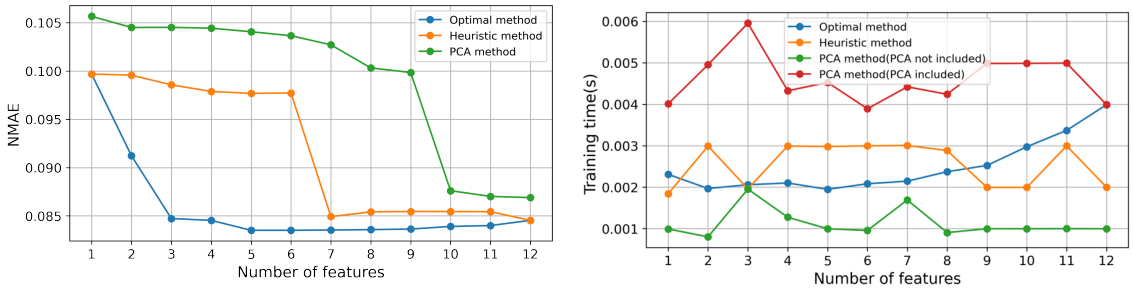
   The plot is shown in Figure 10(a) together with other methods. As the number of principle components increases, the error decreases and the lowest error is achieved when $k = 12$. When k changes from 9 to 10, the error has dropped significantly, and then the decline is very small. We can infer that the first 10 principal components contain most of the output-related information.

2. Measure the training time of the linear regression model for each k. Produce a plot that shows the error values in function of the training time. Use this plot to explain the trade off between estimation error and computational overhead.

   The first thing to point out is that when we change the value of k in a loop, some of the results of the training time become 0. This may be the result of python's optimization of the loop (eg. parallel computing). So we manually change the value of k and record two types of training time: including and excluding PCA calculation time.

Theoretically, when k becomes larger, the training time ignoring PCA calculation time should increase with the increase of the input feature dimension, but there isn't any increasing trend in the experimental results. We think there are two reasons. The first is that due to the small amount of calculation, the training time is greatly affected by other factors (such as the utilization of computer CPU). On the other hand, the calculation of the linear model uses optimization so that the training time does not monotonically increase with the increase of the number of features.
To get the lowest error and training time, it is best that the point on the graph is as close as possible to the origin of the coordinate, then we should choose $k = 12$.

3. 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.



(a) Plot of linear regression NMAE with the number of features or principle components

(b) Plot of linear regression the training time with different number of features using different kinds of method

Figure 10: Plots for 4.3 and 4.4

The role of PCA is to use low-dimensional data to express the information of high dimensional data as much as possible. However, this information may not be related to our prediction target value, in other words, we include some information in the principal components that will worsen the linear prediction, which makes the PCA result worse than the previous two methods. This is clearly shown in Figure 10(a).

4. Produce three plots plots showing the trade off between accuracy and computational overhead for the two feature selection methods studied in Task 3 and Task 4.

   The plot is shown in Figure 10(b). The training time of the optimal method is the average training time under the same number of features, and a loop is used (according to the previous analysis, the optimization of the loop will reduce the results of training time).
   The advantage of PCA is that it uses normalized data to calculate, so that the calculation overhead will be reduced. When we do not consider the calculation time of the PCA itself, it can be seen from the plot that its calculation overheadis the smallest, but when we take the calculation time of the PCA into consideration, it has the longest training time.
   In general, for the question of whether PCA saves computational costs, we should experiment on larger and more complex data sets.

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

The objective of this task is to use a neural network regressor to estimate a service metric Y from device statistics X. The training and evaluation of the model makes use the Python package Keras.

1. Model Training - configure a neural network and train a model M with the training set.
   We firstly use standardization to pre-process the features of X and then separate the data into a

training set and test set with the proportion of 7:3 and then build a simple neural network composed of one input layer with 12 nodes and one output node. Other parameters are further explained in next question.

2. Search the space of hyper-parameters for your model. Identify these hyper-parameters.

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

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

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

   we use a 5 fold cross-validation grid research to set the hyper-parameters one by one. Firstly, we search the parameters in: learning rate[0.01, 0.001], epochs[20, 40], batch size[4, 8, 16]. The other parameters are: L^2 norm regularizer($\lambda = 10^{-5}$), activation function: relu. The scoring method is "neg_mean_absolute_error" which is the negative mean absolute error because the default setting is to maximize the score while we want to minimize the error.
   The best result with a score of -1.764 comes from the parameters [learning rate: 0.01, epochs: 40, batch size: 16].
   Then we search parameters in L^2 regularizer $\lambda[10^{-3}, 10^{-4}, 10^{-5}]$ and L^1 regularizer $\lambda[0.05, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}]$, and the the best result with score -1.678 comes from the parameters L^1 regularizer $\lambda = 0.05$.
   Then we add another hidden layer with the same configuration of the input layer to the network. We only adjust the number of the nodes, and the best result with a score of -1.654 is achieved with 6 nodes. Then we add another hidden layer but the performance can't be improved so we stop here.
   The optimized parameters are:

   (a) Optimizer: learning rate 0.01, number of epochs 40, batch size: 16

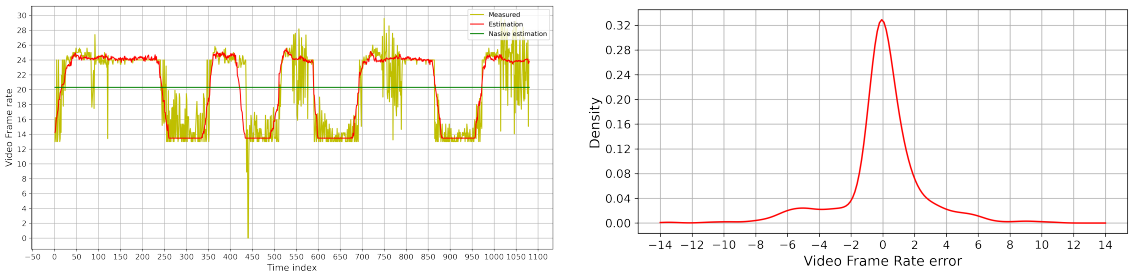
   (b) Regularizer: L^1 norm, $\lambda = 0.05$;

   (c) Architecture: one input layer with 12 nodes, one hidden layer with 6 nodes and one output node, activation function: relu

3. Accuracy of Model M - compute the estimation error of M on the test set.
   NMAE of the linear model and the naive estimation are 0.075 and 0.22.

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.

5. Produce a density plot of the estimation errors $y_i - \hat{y}_i$ using the test set.
   The results are shown in Figure 11.



(a) Measurements, neural network estimations and naive estimation for the Video Frame Rate values

(b) Density plots and histograms of the Video Frame Rate error in the test set

Figure 11: Plots for 5.4 and 5.5

6. Based on the above evaluation, discuss the accuracy of estimating the Video Frame Rate. Discuss the role of hyper-parameters search for obtaining an effective model. Compare method and results with linear regression in Task II.
   For neural networks, there are many hyper-parameters that need to be tuned. Generally speaking, we cannot grid search all parameters at once, but adjust different types of parameters step by step. And we cannot guarantee that the parameters we use are optimal, thus we can only optimize the parameters as much as possible under the constraints of time and computing resources.
   Compared with the linear regression with a minimum NMAE 0.084 shown in Task 3, the neural network approach achieves better results with NMAE 0.075. Compared with Figure 5, the fluctuation of the prediction in Figure 11 is smaller, and the probability distribution of the error is more concentrated on the zero point. But the disadvantage of the neural network is that computation overhead is very huge. In this project, the training time of the linear regression model is in milliseconds level, while the training time of the neural network takes several minutes.

Discussion and Conclusion

In this project, we mainly use a linear regression model and neural network to solve a supervised learning problem related with video-on-demand network services. The prediction result of the neural network is better than the linear regression model, but its disadvantage lies in the huge amount of computation and the complicated parameter tuning process.
There are two main areas for improvement in the experiment. One is that in the analysis of PCA, the training time is too short, which makes it difficult to judge the impact of PCA on the computation time. A larger data set should be used to discuss this problem. The second one is that in the training of neural networks, there is still a lot of room for improvement in the hyper-parameter tuning. Not only can the range of grid search be increased, but random search can also be used. It is also worthwhile to consider different network structures and activation functions.