Contents

[Problem identification 2](#_Toc71549110)

[Problem Statement 2](#_Toc71549111)

[Context 2](#_Toc71549112)

[Criteria for Success 2](#_Toc71549113)

[Scope of Solution Space 2](#_Toc71549114)

[Constraints 2](#_Toc71549115)

[Stakeholders 2](#_Toc71549116)

[Dataset Description 3](#_Toc71549117)

[Customer Transaction Data 3](#_Toc71549118)

[Data Wrangling 3](#_Toc71549119)

[EDA 4](#_Toc71549120)

[Scatter Plots: features against timestamp 4](#_Toc71549121)

[Correlation Matrix 13](#_Toc71549122)

[Pairplot 14](#_Toc71549123)

[Modeling 15](#_Toc71549124)

[Hyperparameter Tuning 15](#_Toc71549125)

[Feature Importance 17](#_Toc71549126)

[Feature Aggregation 17](#_Toc71549127)

[Conclusion 18](#_Toc71549128)

[Recommendation 18](#_Toc71549129)

[Future Scope 18](#_Toc71549130)

# Problem identification

## Problem Statement

Given an IT department’s operational data for multiple software machines, is it possible to predict run time anomalies to investigate potential failures and prevent downtimes before they happen.

## Context

The dataset consists of metrics measured from the operating system and from WebLogic Server monitoring beans for 200 operational machines. We want to apply machine learning to detect anomalies and prevent failures which can then be used by software companies for valuable cost savings.

## Criteria for Success

* Using machine learning, we are successfully able to categorize data as regular and anomalous
* Set reliable thresholds to predict anomalies

## Scope of Solution Space

Analysis is restricted to data from part 67.

## Constraints

* Data only available for 2.5-month span
* Subject matter expertise required to understand features
* Potentially many highly correlated features

## Stakeholders

* VP Information Technology
* Engineering Director
* VP Finance

# Dataset Description

The data is taken from Kaggle and consists of 200 datasets representing separate software machines. We will be looking at machine 67, with the following columns:

## Customer Transaction Data

Transaction data for all customers for duration of campaigns in the train data. It consists of 235 columns with information like:

* Active connections
* DB connection activity
* Connection delay
* Daemon thread count
* Failed wait for connections
* Heap usage activity
* Memory space activity
* Physical mem activity
* Rel. unavailable connections
* Reserve request
* Stuck threads
* Swap activity
* System CPU
* Thread CPU time
* Thread User time
* Total thread count

# Data Wrangling

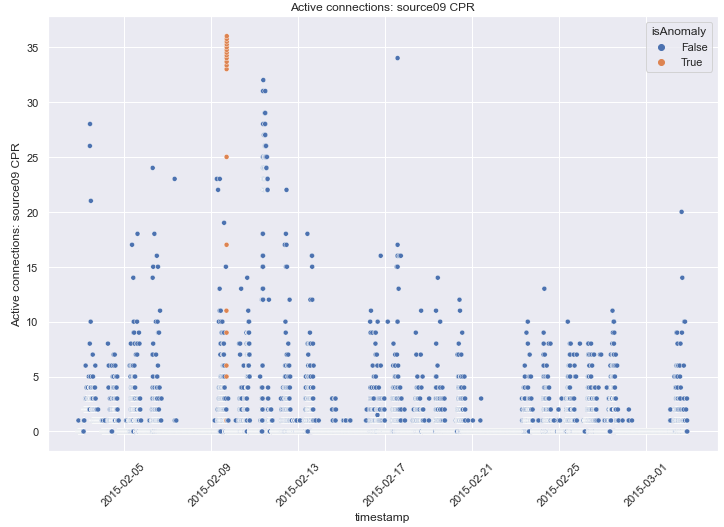
1. We start by dropping all columns that do not contain any unique value.
2. We change column names to a more readable format.
3. We change the data type of ‘timestamp’ column from string to datetime.
4. Lastly, we drop columns with correlation of greater than 0.5 or less than -0.5.

We end up with 43 columns/variables df1and save this dataset in our folder.

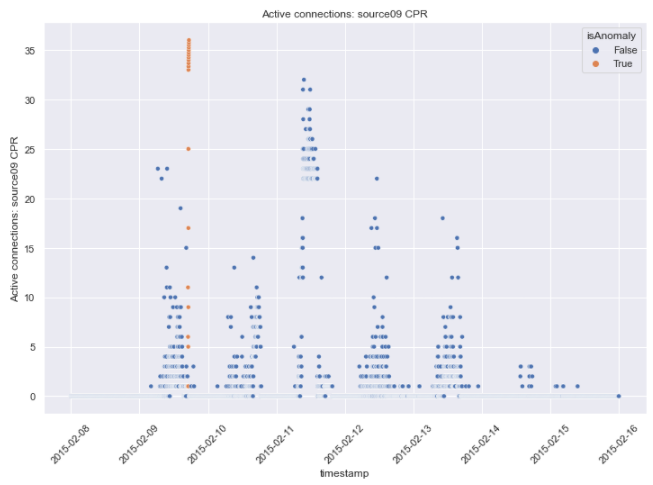
# EDA

## Scatter Plots: features against timestamp

### Active connections: source09 CPR (entire range)

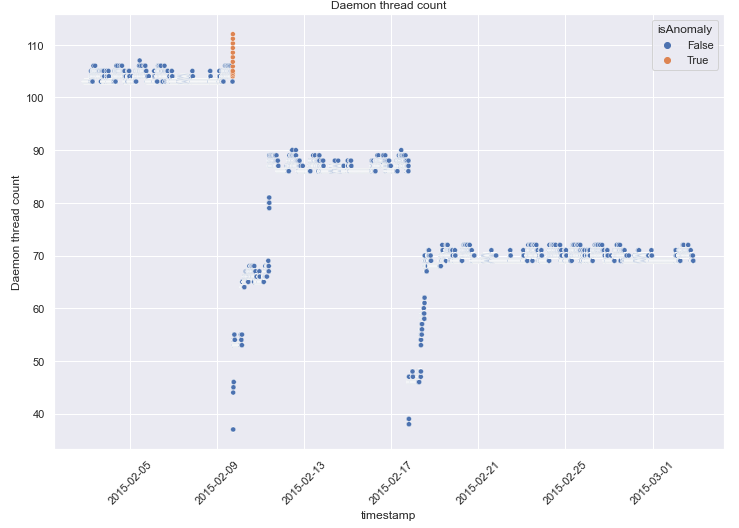


###### Active connections: source09 CPR (week of anomalies)

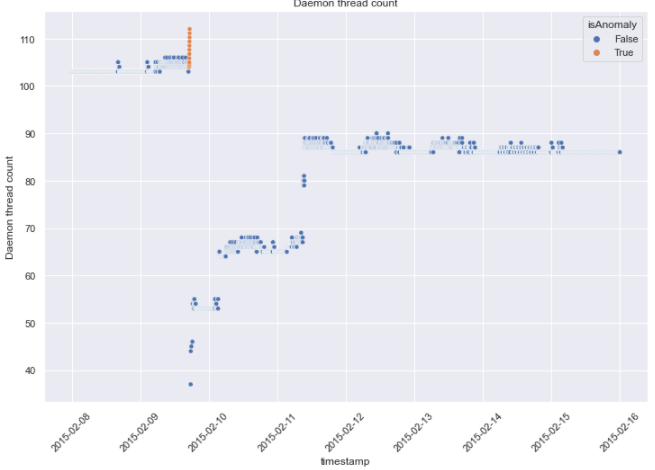


There is a spike in active connections on the day of anomalies. Maximum active connections occur during the work hours. As seen in the second graph, we see little to no activity on 2/9 (Sunday) and 2/15 (Saturday). We also see a spike in activity on Wednesday after the downtime. This is not true for other Wednesdays.

### Daemon thread count (entire range)

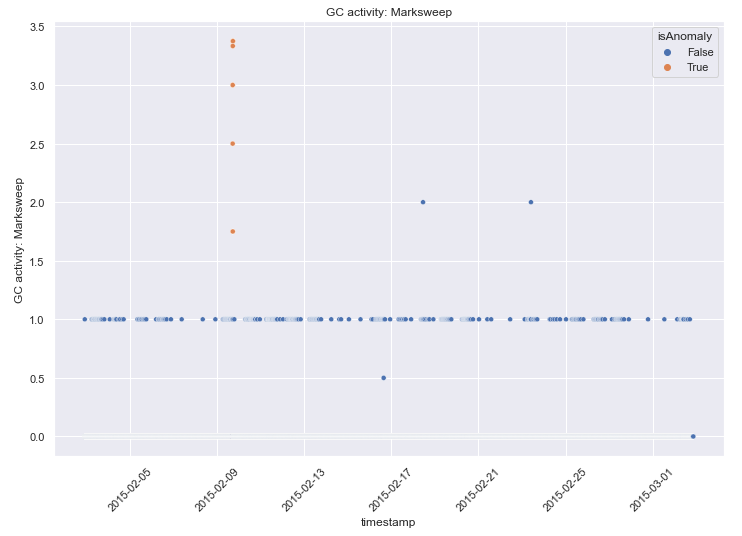


###### Daemon thread count (week of anomalies)

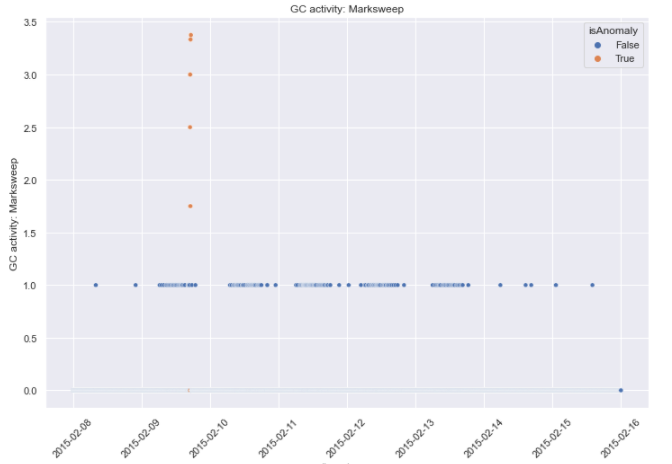


We observe a spike in Daemon thread count on the day of anomalies. Again, we see maximum Daemon thread counts during the work hours. Daemon counts are low the day after recorded anomalies.

### GC activity: Marksweep (entire range)

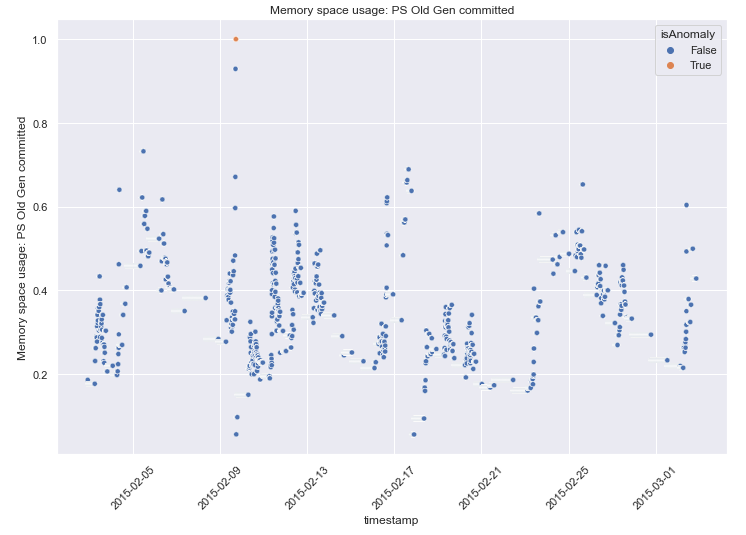


GC activity: Marksweep (week of anomalies)

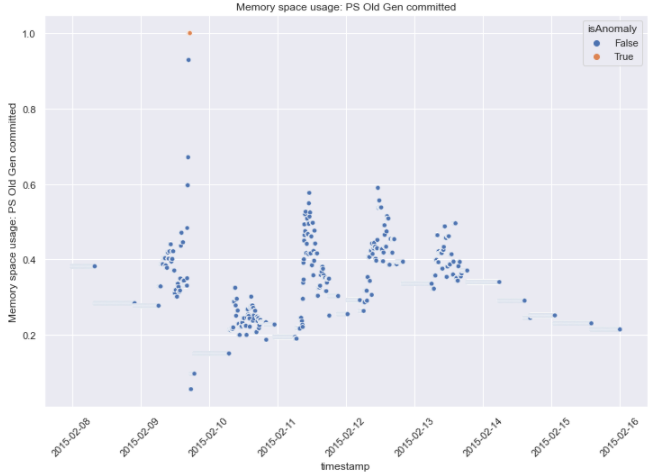


There is a spike in GC activity: Marksweep on the day of anomalies. GC activity: Marksweep is highest during the work hours with minimal activity during nights and weekends.

### Memory space usage: PS Old Gen committed (entire range)

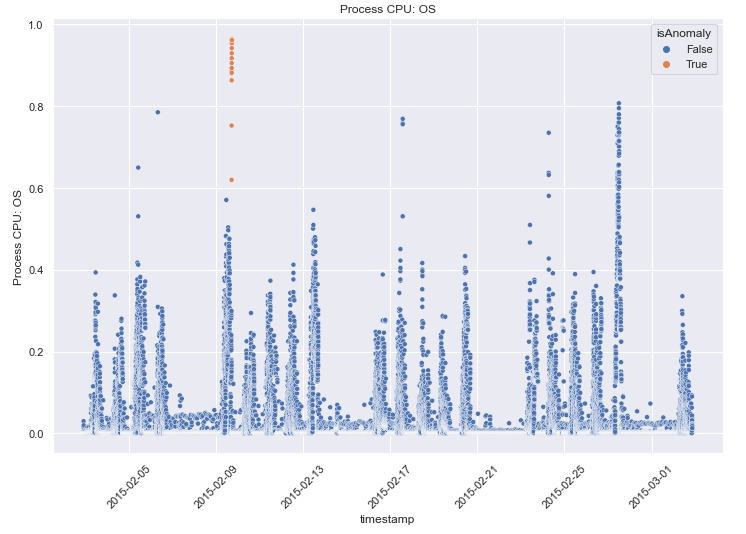


Memory space usage: PS Old Gen committed (week of anomalies)

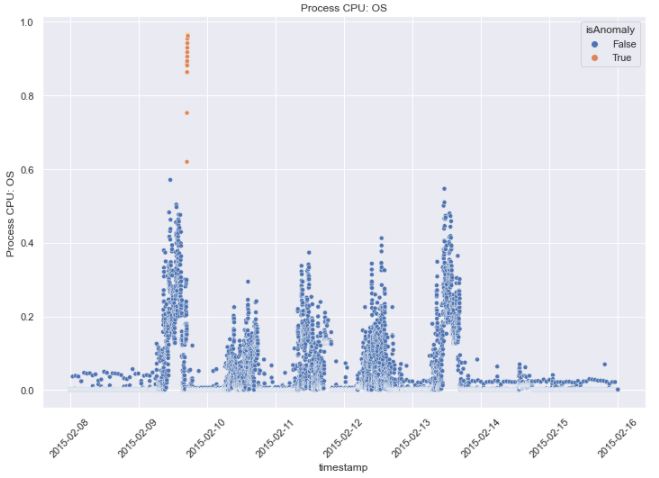


We see an anomaly when the memory space usage reaches 100%. Memory space usage seems to be maximum on Mondays.

### Process CPU: OS (entire range)

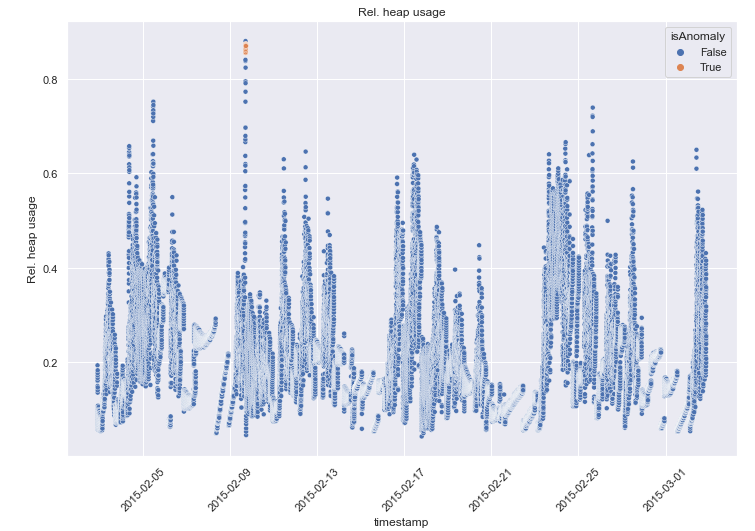


Process CPU: OS (week of anomalies)

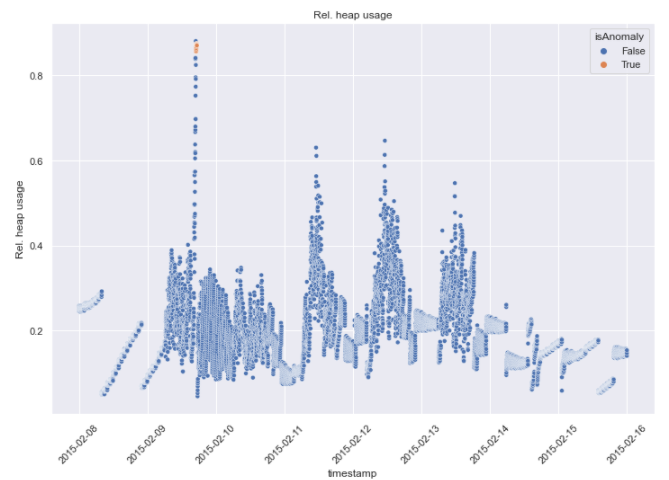


We start seeing anomalies when the process crosses ~ 0.85 and the system seems to recover after dropping below 0.6. There are other days when the process crosses 0.8, without failing.

### Rel. heap usage (entire range)

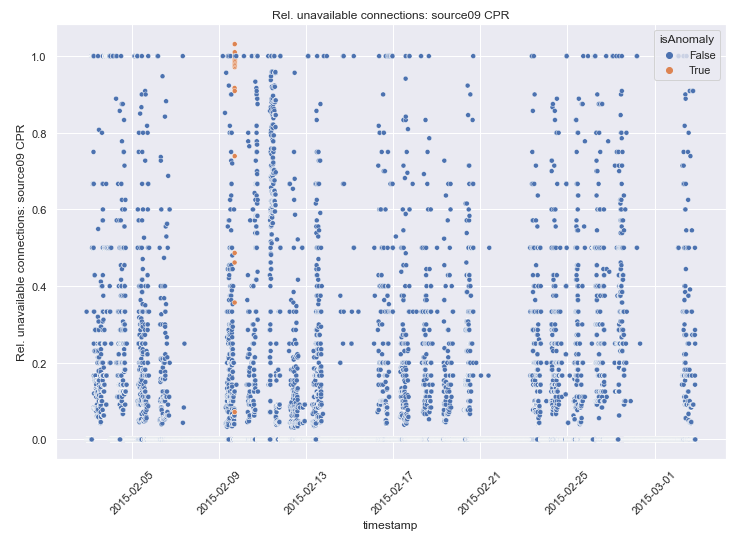


Rel. heap usage (week of anomalies)

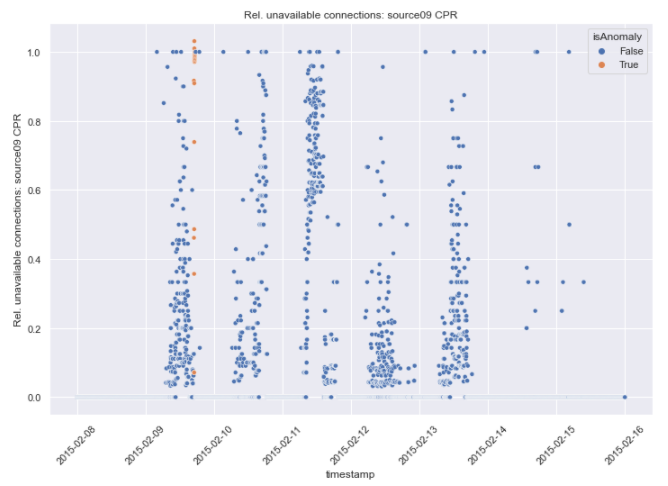


We see anomalies after multiple consecutive spikes over 0.8 on Rel heap usage. It should be noted that we see consecutive spikes on other days that don’t result in anomalies. However, these spikes on other days do not cross 0.8 usage.

### Rel. unavailable connections: source09 CPR (entire range)

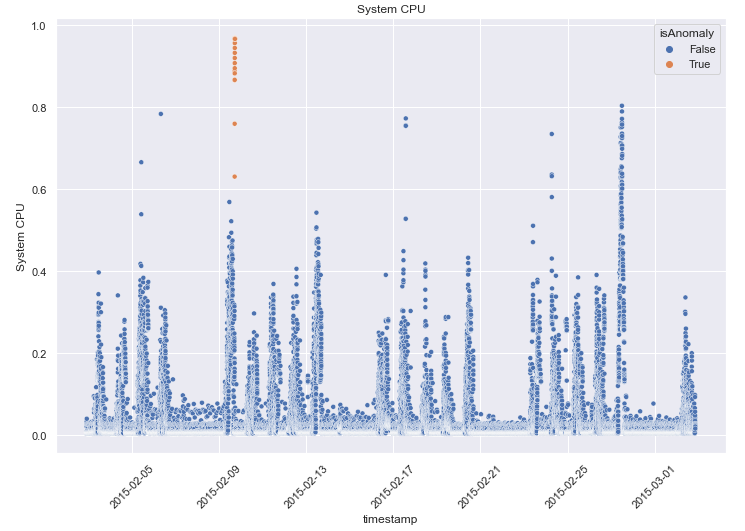


Rel. unavailable connections: source09 CPR (week of anomalies)

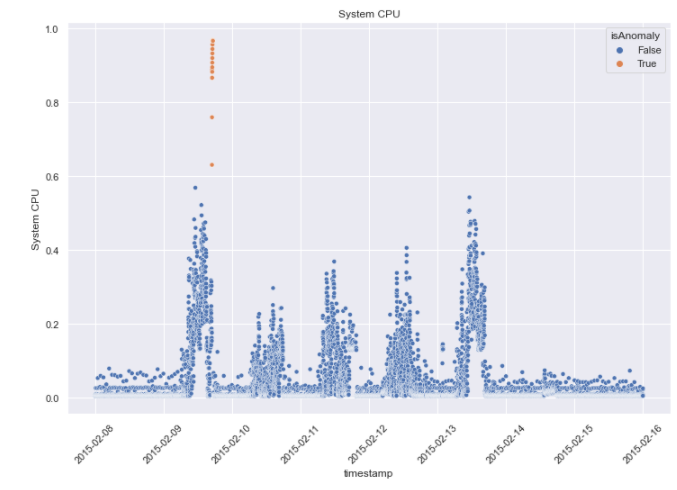


This features reaches 1 on a number of occasions, but anomalies are seen only when this feature crosses 1.

### System CPU (entire range)

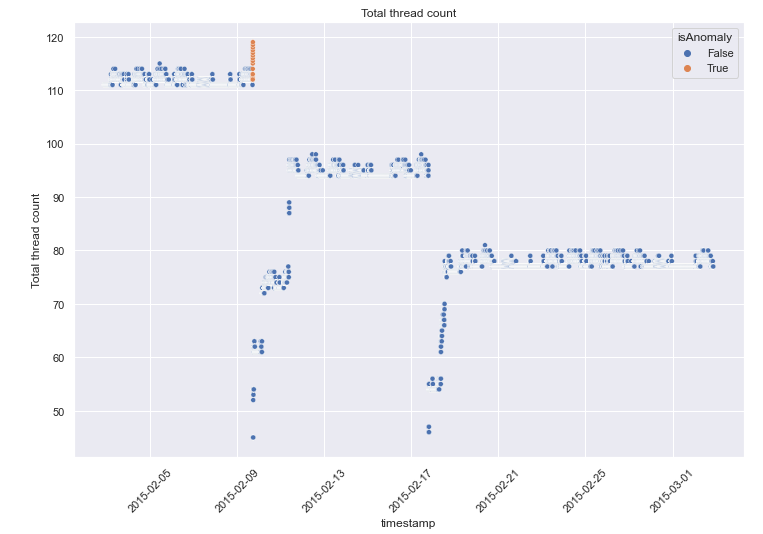


System CPU (week of anomalies)

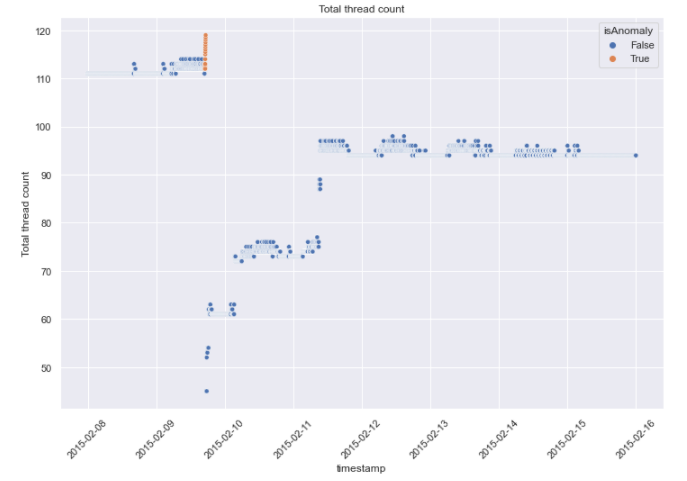


We start seeing anomalies when the feature crosses ~ 0.85 and the system seems to recover after dropping below 0.6. There are other days when the process crosses 0.8, without failing.

### Total thread count (entire range)

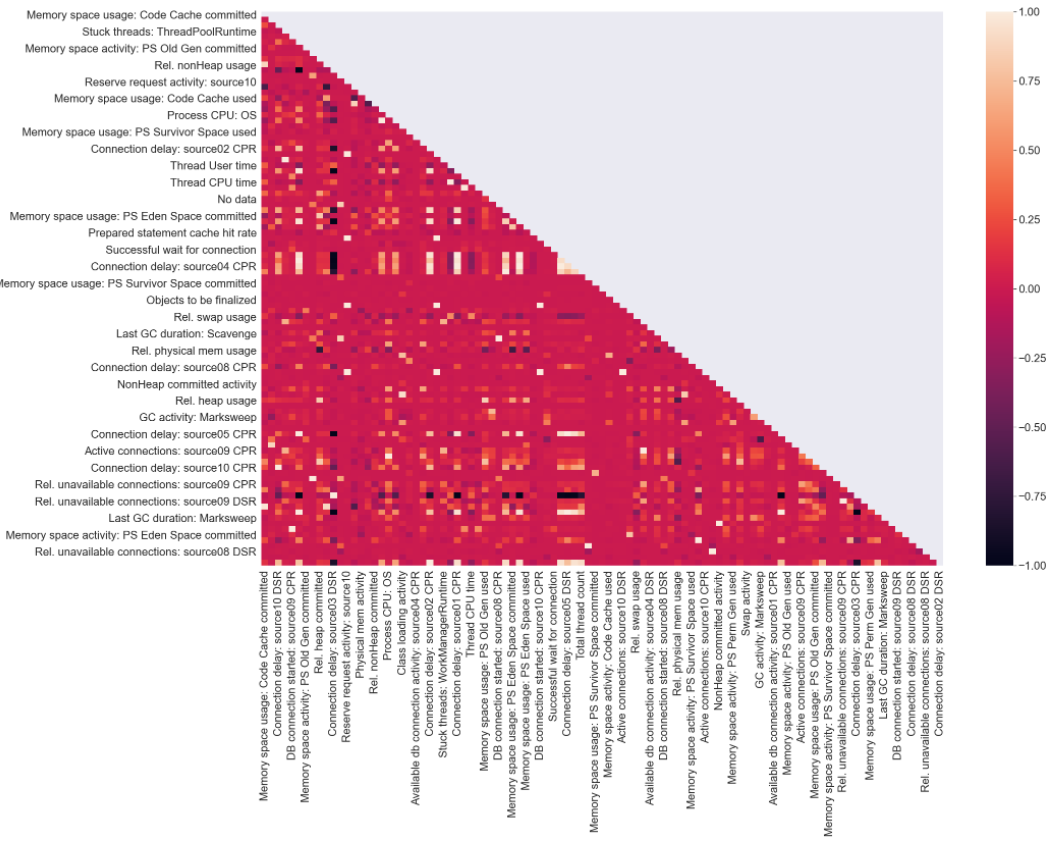


Total thread count (week of anomalies)



We see total thread count crossing 115 on the day of anomalies. We see maximum total thread counts during the work hours. Total counts are low the day after recorded anomalies.

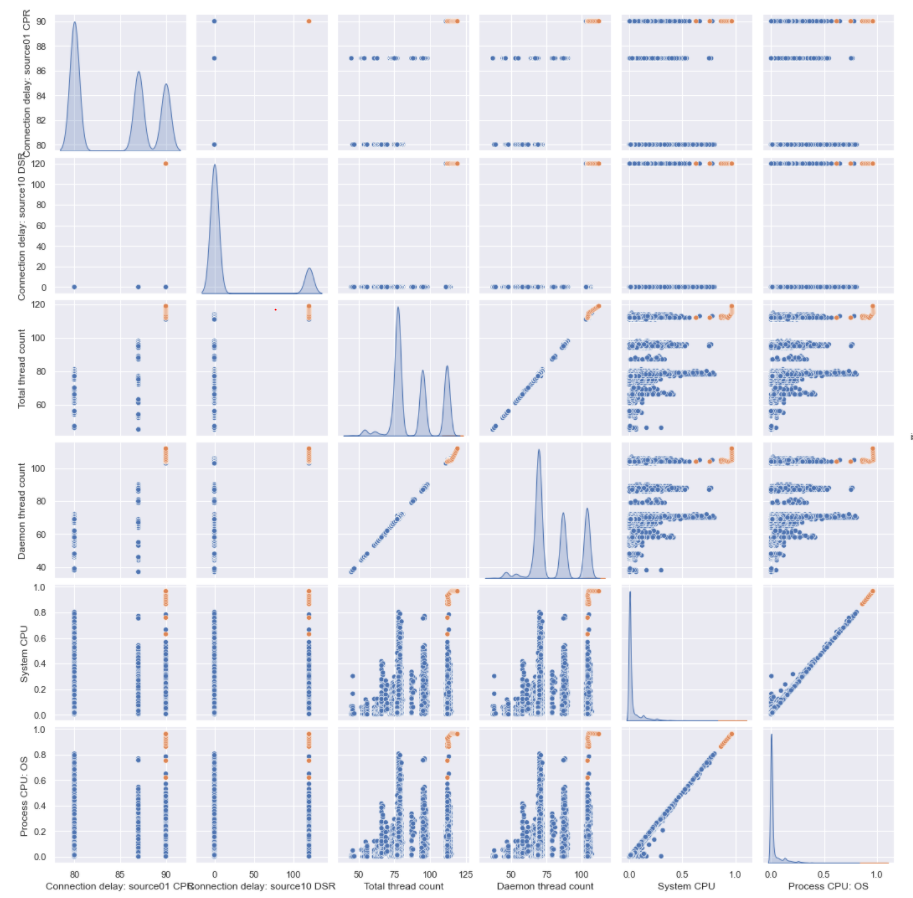
## Correlation Matrix



From the correlation matrix there are multiple features with a moderate to very high degree of correlation with each other.

## Pairplot

Here is a pairplot of some of the highly correlated features:



* System CPU and Process CPU: OS are highly correlated
* Daemon thread count and Total thread count are highly correlated
* Connection delays are highly correlated
* When thread counts spike, system and process cpu also spike
* When thread counts spike, delays also spike

# Modeling

We started by separating the dataset into train and test sets. We checked multicollinearity between all features and removed redundant columns.

We then fitted the test data to three models and compared their performances. Our metric of choice was AUC-ROC curve. Here are our calculations:

|  |  |
| --- | --- |
| Model | Area under ROC curve |
| Random Forest | 1 |
| Logistic Regression | 0.49 |
| XGB Classifier | 0.875 |

We decide to go with the XG boost model and proceed with tuning the model hyperparameters.

# Hyperparameter Tuning

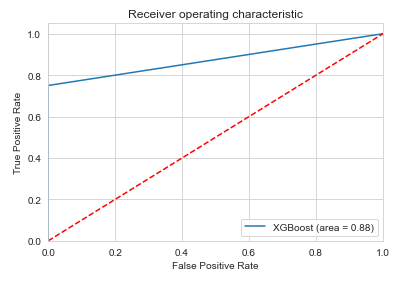
We selected the following parameter grid to use with GridSearchCV:

|  |  |
| --- | --- |
| eta | 0.01, 0.05, 0.1 |
| max\_depth | 9, 10, 11 |
| scale\_pos\_weight | 49, 50, 51 |
| max\_delta\_step | 4, 6, 8 |
| grow\_policy | depthwise, losswise |

This gave us a best score of 0.86 for the following parameters:

|  |  |
| --- | --- |
| eta | 0.1 |
| max\_depth | 10 |
| scale\_pos\_weight | 1 |
| max\_delta\_step | 4 |
| grow\_policy | depthwise |

We used these hyperparameters in our XGB Classifier model, fit it to our training data and predict the test data labels. We get a roc\_auc score of 0.88:



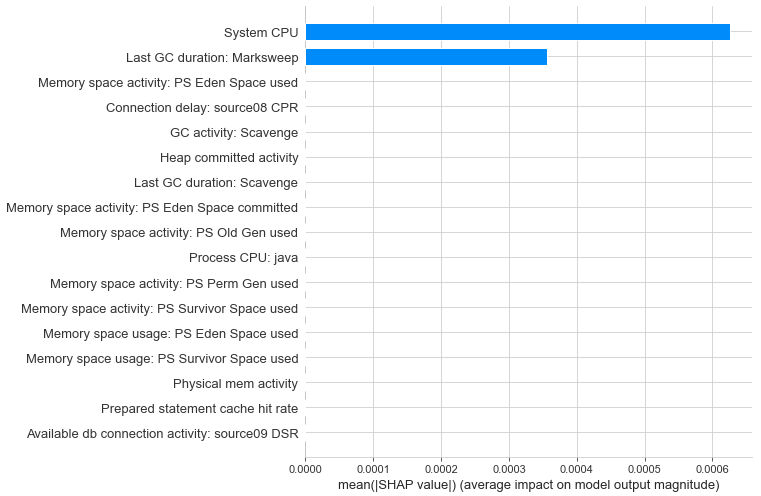
This is the confusion matrix we obtained:



**F1 score**: 0.86

# Feature Importance

At the end, we performed feature importance analysis using the SHAP module:



The graph above show us that the most important features that contribute to anomaly detection are:

1. System CPU
2. Last GC duration: Marksweep
3. Memory space usage: PS Eden Space used
4. Connection delay: source08 CPR

# Feature Aggregation

Finally, we aggregated the four most important features to determine the cutoff points. For doing this, we calculated 95th percentile of each of the four features from the training setto calculate the threshold. We obtained the following cutoff limits:

|  |  |
| --- | --- |
| System CPU | 0.19 |
| Last GC duration: Marksweep | 1897 |
| Memory space usage: PS Eden Space used | 0.56 |
| Connection delay: source08 CPR | 61 |

# Conclusion

The biggest indicators of anomalous behavior in software operational data are ‘System CPU’, ‘Last GC duration: Marksweep’, ‘Memory space usage: PS Eden Space used’, ‘Connection delay: source08 CPR’. The software will most likely fail soon after the system approaches these threshold values.

## Recommendation

To minimize future failures, we recommend implementation of an alert mechanism to be implemented that notifies stakeholders as soon as the cutoff limit of any of the 4 most important features is reached. The stakeholders can then troubleshoot to determine the underlying cause and prevent operational failures.

## Future Scope

We recommend continuing to record this data to refine the threshold limits to get more reliable values. We also recommend applying this analysis to remaining software machines for additional cost savings.