Analyze historical performance metrics to predict critical system errors before system degradation

Anthony Pagan

12/13/2020

Introduction

Many organizations rely on monitoring systems to monitor performance of their backend systems. The expectation is that the monitoring will notify support staff when critical systems are in a critical state. Unfortunately, many time support teams are only alerted after a system is degraded. In some cases, manual monitor tuning results in false alerts.

The main goal of this paper is to build a proactive monitoring system model based on performance metric calculations that can be correlated to events and allow teams to build automated resolutions. Such a system would reduce support cost and increase efficiency while minimizing false non-actionable alerts.

Methodology

There are several journals with different approaches to proactive monitoring that include some self-healing mechanisms to resolve system performance issues. Some methods include existing algorithms and some attempt to build their own. The existing algorithms include multiple linear regression, LSTM, RNN, Autoregression ARIMA models, ANN as well as others.

LSTM is a combination of a long and short-term memory RNN. A RNN or recurring Neural Network based on LSTM is called an LSTM neural network. The subtle difference of LSTM vs RNN is the added forget gate in addition to the input and output gate[1] which is a pluse when using sequence data. LSTM can use auto-encoder to capture the temporal structures from monitoring data and uses deep neural network to determine reach state [5]. LSTM can also be used as a many to one network with requests-to-vector. It applies RNN-LSTM network to predict the performance and workload of servers [8].

The Monte Carlo approach [2] uses long term non-stationary data and uses residual short-term process to capture both the long-term linear combination of certain basis function with random application evolving with time, while short term is modeled as a traditional Auto regression process. Parameter for long-term and short-term data are estimated using sequential Monte Carlo (SMC). This approach is similar to time series prediction framework for PRESS, AGILE, ARIMA, NARNN, LSTM and BLSTM for resource usage prediction, analysis and comparison of performance of proposed methods [11]. It uses different method for selection of relevant subset of features and prediction and stability-based joint framework for selection of suitable feature selection techniques.

Other options include decision trees, k-means and gaussian clustering transformed with PCA [6][10]. First each, a metric is scaled to zero and a unit of variance, then principal component analysis is used to transform the metrics to get top 3 dimensions.

For this paper we will use several of these methods. Something similar to a Monte Carlo approach will be needed because of the type of dataset and our objectives. We have time series data and are trying to predict a binary variable where 1 = Ticket and 0 = No ticket. We have several options to work with binomial prediction, we will run several , compare and select the top performers. We will compare linear regression,

LASSO, SVM, CTREE and Ridge Regression performance then compare the top performers to Keras NN to derive our final model. To impute any missing data we need a Time series method. The na_kalman function will be used similar to ARIMA to impute our partial missing time series that become NA as a result transformation. In addition, we will use BoxCox and LDA to finalize our independent variables.

Data Description and Exploration

In this paper we are using a monitoring dataset in 2 files which contain performance data from several development servers. Both datasets are related by time stamp and managedentityGUID. One dataset includes alert and the severity of the alert. The severity helps determine if an alert would generate a ticket to the support groups. The second file includes the performance metrics. Combining the dataset allow us to know the value of each performance metric when an alert is servere enough to require some action and generate a ticket.

```
setwd(getwd())
d1<- as.data.frame(read.csv('../Datasets/TestQuery.csv', header = TRUE,quote = "\"",sep=','))
cnames<-c('AlertName','AlertDescription','Severity','Priority','DateTime','ManagedEntityGuid')
d2<- as.data.frame(read.csv('../Datasets/TestQuery3.csv',header= FALSE, col.names = cnames, quote = "\"

joineddata<-right_join(d1, d2,  by = c('DateTime','ManagedEntityGuid'))
joineddata<-subset(joineddata,!is.na(SampleValue))%>%
  subset(select=c(-AlertName.x,-Severity.x))%>%
  rename(AlertName = AlertName.y, Severity = Severity.y)%>%
  mutate(ServerName = case_when
         ((ManagedEntityGuid == '2A909362-BE1E-F148-835C-96787AE2775E') ~ 'Server1',
         (ManagedEntityGuid == '6EA459BD-0616-71F6-FC4A-7989A2DCF9FD') ~ 'Server2',
         (ManagedEntityGuid == 'A1A8A538-8B75-B8F5-5F44-7159F192F602') ~ 'Server3',
         (ManagedEntityGuid == 'AF487CE3-8C62-2C7A-F748-9241D73F0403') ~ 'Server4',
          TRUE ~ 'Server5'))%>%
  mutate(Ticket = case_when
         ((Severity == 2) ~ 1,
           TRUE ~ 0))
nm1<-names(d1)
nm2<-names(d2)
```

The columns for each dataset are listed below.

Dataset 1:

DateTime, ManagedEntityGuid, Objectname, CounterName, SampleValue, AlertName, Severity

Datset 2:

AlertName, AlertDescription, Severity, Priority, DateTime, ManagedEntityGuid

We begin by merging the 2 dataset based on DateTime and ManagedEntityGUID, and remove any rows with NA for sample values. In addition, we create 2 columns. The servernames columns gives us a readable name for each managedentityguid. The ticket column is generated by using the serverity column. Any alert with a serverity of 2 would be a critical severity and would therefor initiate a ticket to the support groups.

Joined Column Description:

	Data_Description
DateTime	Date stamp of alert
ManagedEntityGuid	Identity of monitored object
Objectname	Name of component
CounterName	Performance Metric Name
SampleValue	Value of Metric
AlertName	Alert Generated
AlertDescription	Description of Alert Generated
Severity	Severity of Alert
Priority	Priority of Alert
ServerName	Name of Server
Ticket	Was a ticket generated

A glimpse of the data show the initial joined dataset has some columns that may require transformation. DateTime should be date while Severity, Priority and Tickets may be more useful as factors. We can also eliminate the Objectname and Alert Description columns as their details are related to Countername and Alert columns respectively.

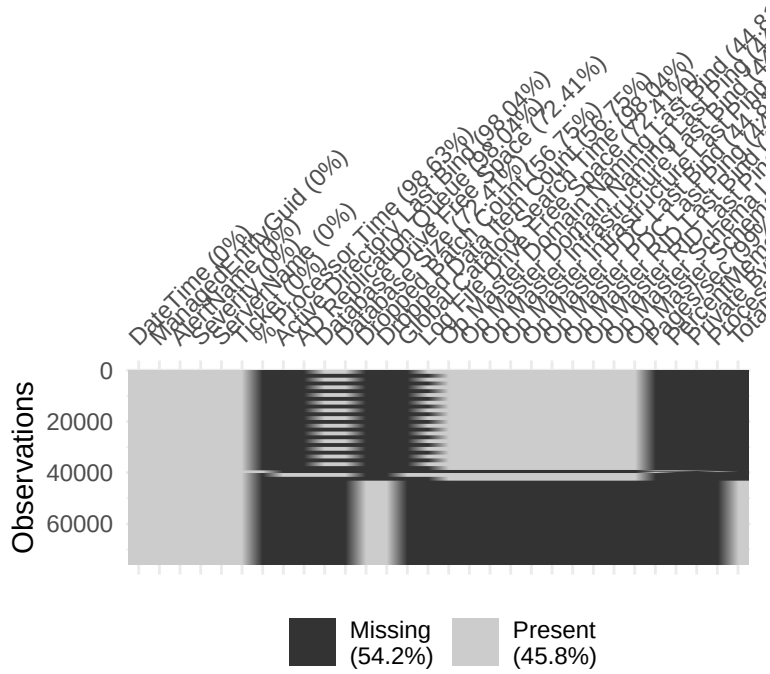
The table comparison of ticket to Servername show server3 has the bulk of the performance metrics data.

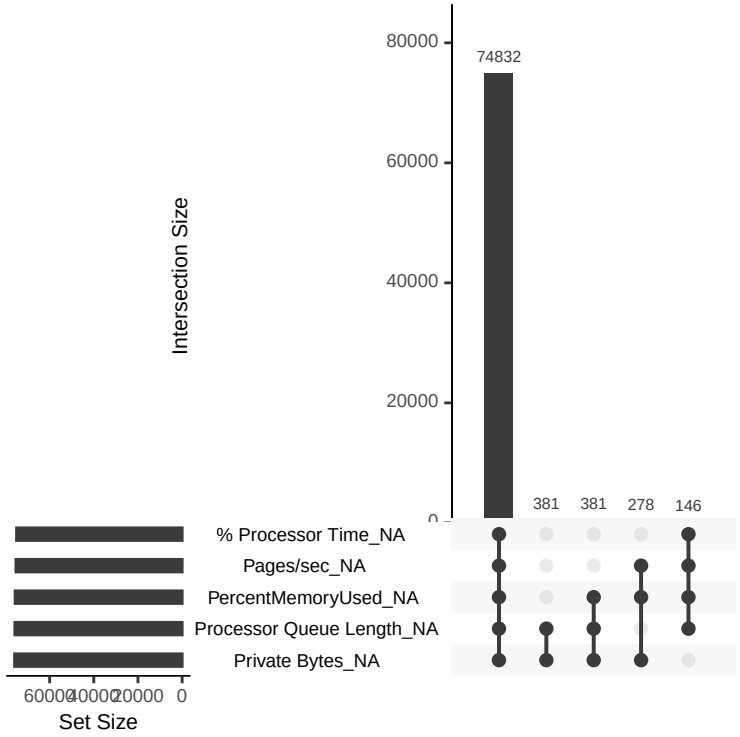
```
## Rows: 588,162
## Columns: 11
## $ DateTime          <fct> 11/10/2020 14:45, 11/10/2020 14:45, 11/10/2020 14...
## $ ManagedEntityGuid <fct> A1A8A538-8B75-B8F5-5F44-7159F192F602, A1A8A538-8B...
## $ Objectname        <fct> PDC Op Master, PDC Op Master, PDC Op Master, PDC ...
## $ CounterName       <fct> Op Master PDC Last Bind, Op Master PDC Last Bind,...
## $ SampleValue       <dbl> 0.001, 0.001, 0.001, 0.001, 0.001, 0.001, 0.001, ...
## $ AlertName         <fct> The Time Skew monitor has failed., The Time Skew ...
## $ AlertDescription  <fct> Unable to contact Root DSE.  The error returned w...
## $ Severity          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ Priority          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ ServerName        <chr> "Server3", "Server3", "Server3", "Server3", "Serv...
## $ Ticket            <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
```

Var1	Var2	Freq
0	Server1	0
1	Server1	98640
0	Server2	0
1	Server2	1905
0	Server3	68954
1	Server3	379247
0	Server4	0
1	Server4	556
0	Server5	38714
1	Server5	146

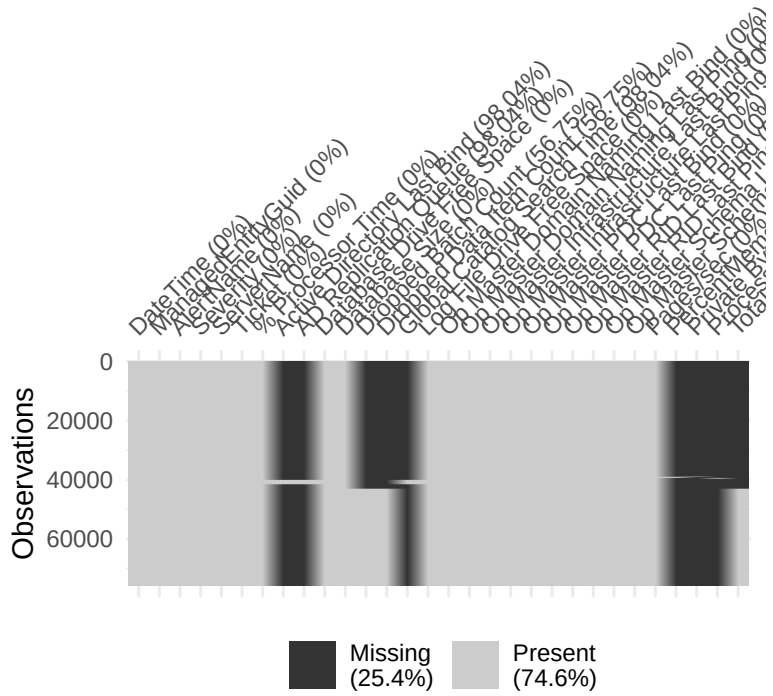
Transformation

In the data transformation we remove Object and AlertDescription columns and spread the dataset by the samplValue and countername. The new form of the data results in a high level of NA counts.



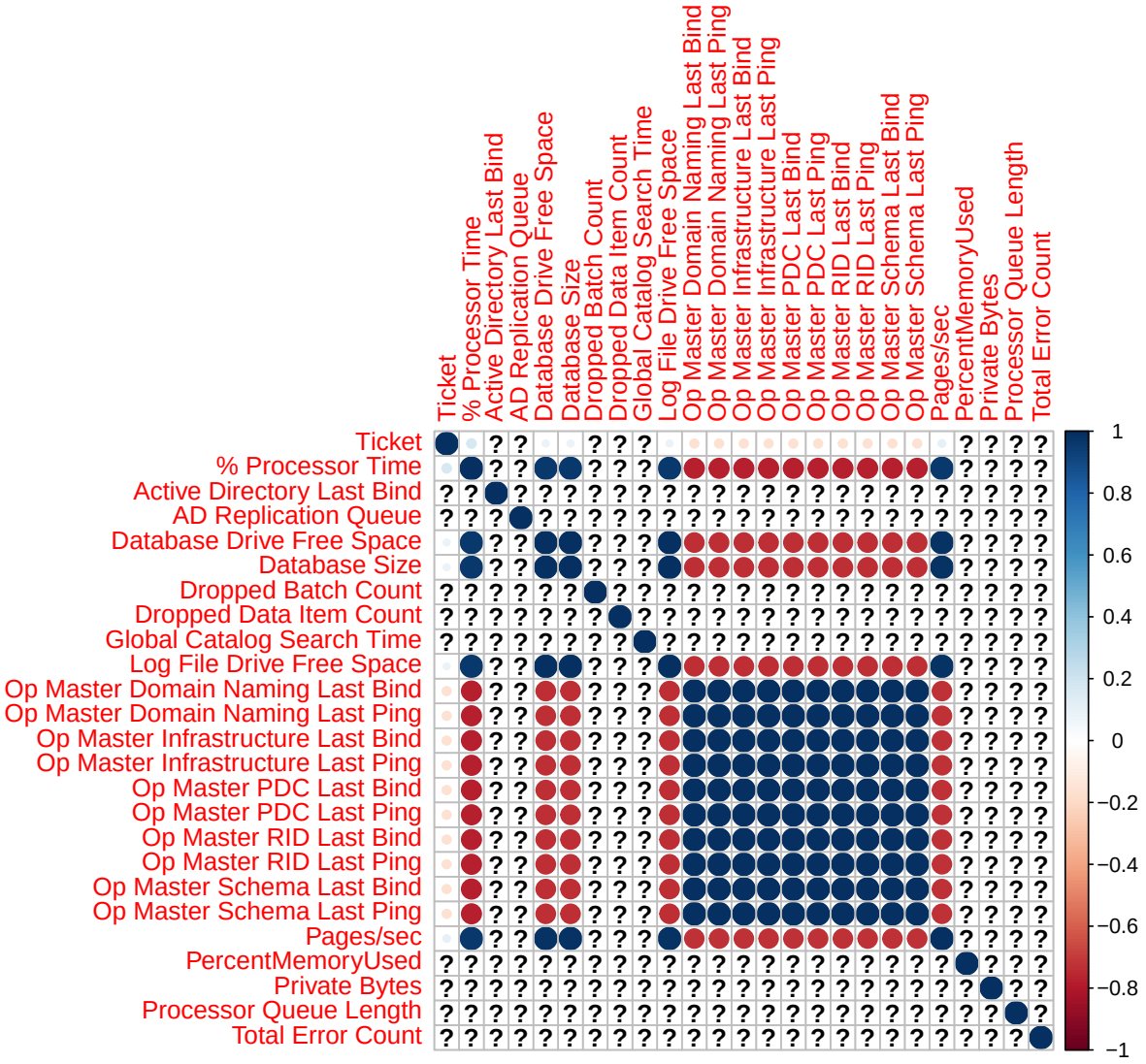


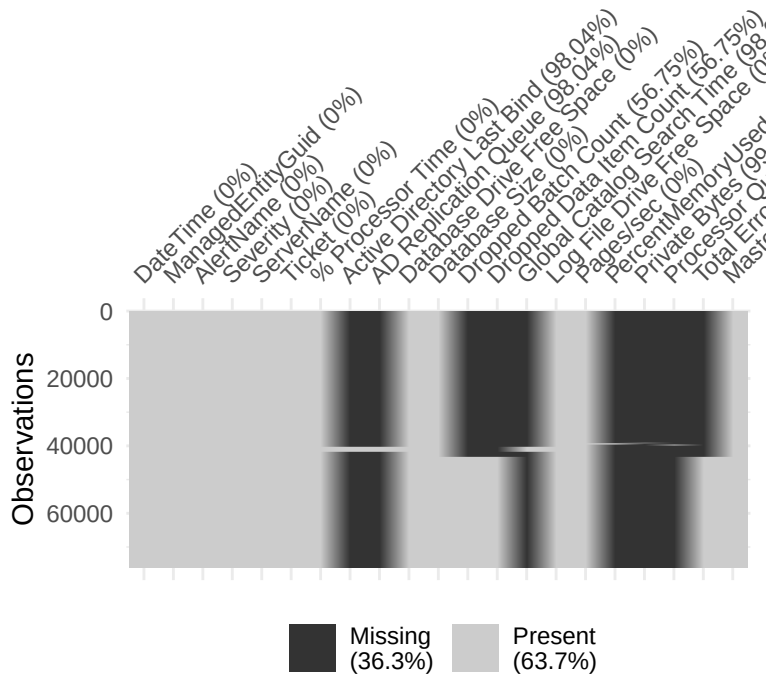
We use the na_kalman function to impute missing data. This function uses Kalman smoothing on structural time series data to impute and estimate missing data. After the transformation the vis_miss function still show a high level of missing data in 9 columns.

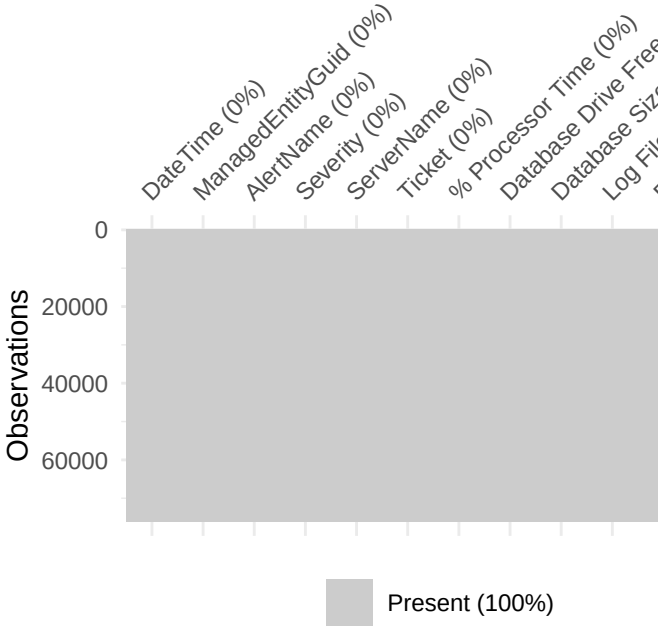


Visualization

Correlation charts show the Op Master* columns are highly correlated. Therefore, we combine the Op Master* columns by rowmeans and name the new column MasterAvg column. The vismis functions still show some columns missing > 98% of their data. We remove any columns with missing data to get our final dataset. At this point the dataset no longer show any missing data.




DateTime (0%)
ManagedEntityGuid (0%)
AlertName (0%)
Severity (0%)
ServerName (0%)
Ticket (0%)
% Processor Time (0%)
Active Directory Last Bind (98.04%)
AD Replication Queue (98.04%)
Database Drive Free Space (0%)
Database Size (0%)
Dropped Batch Count (56.75%)
Dropped Data Item Count (56.75%)
Global Catalog Search Time (98
Log File Drive Free Space (0%)
Pages/sec (0%)
PercentMemoryUsed
Private Bytes (98
Processor Q
Total Erro
Mast
Observations
0
20000
40000
60000
Missing
(36.3%)
Present
(63.7%)


DateTime (0%)
ManagedEntityGuid (0%)
AlertName (0%)
Severity (0%)
ServerName (0%)
Ticket (0%)
% Processor Time (0%)
Database Drive Free
Database Siz
Log Fil
Observations
0
20000
40000
60000
Present (100%)

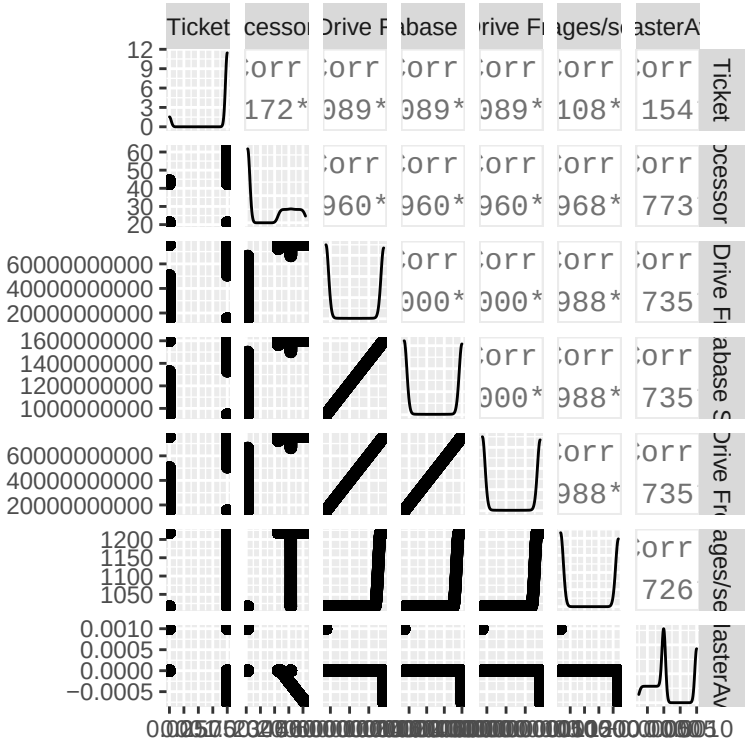

Ticket
cessor
Drive F
abase
rive Fr
ages/se
asterA
Corr
172*
089*
960*
000*
988*
108*
154
773
735
726
Ticket
Drive Fr
ages/se
asterAv

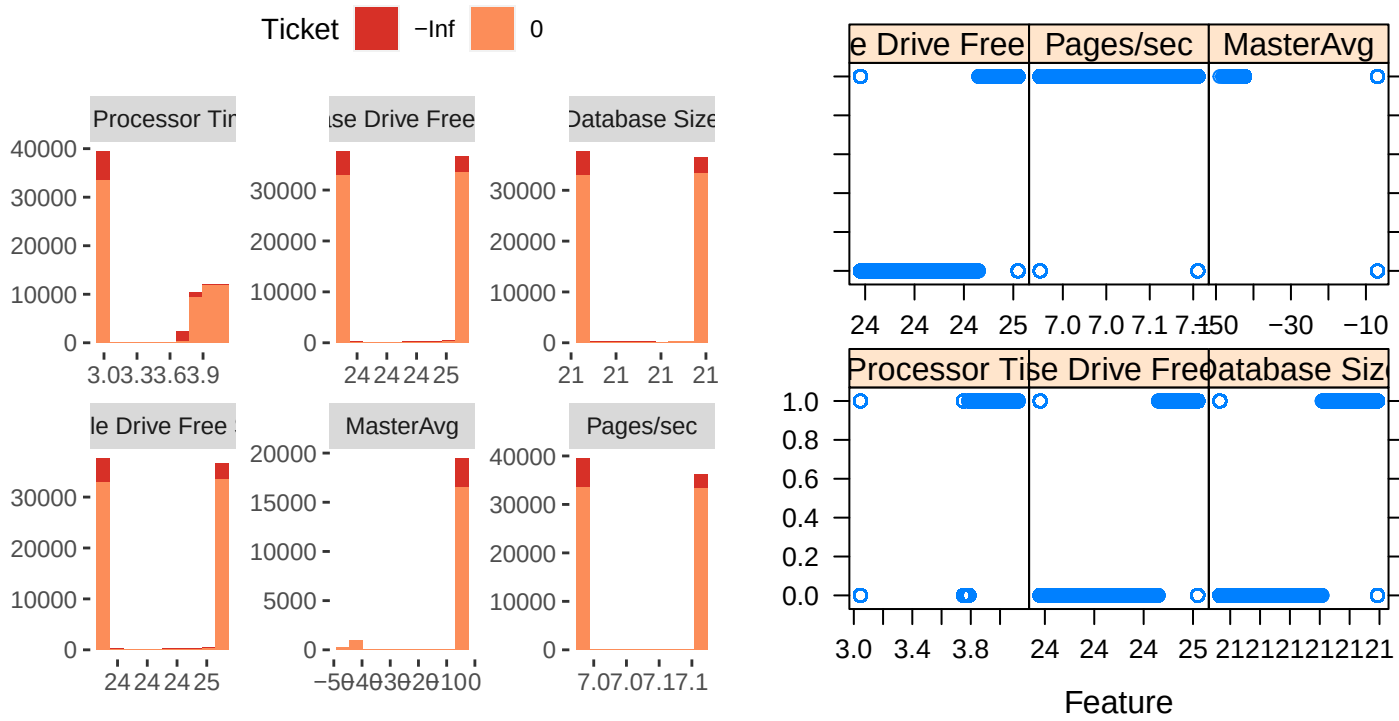
Table 1: Final Dataset 12 columnns

x
DateTime
ManagedEntityGuid
AlertName
Severity
ServerName
Ticket
% Processor Time
Database Drive Free Space
Database Size
Log File Drive Free Space
Pages/sec
MasterAvg

Table 2: Variables recommended for removal at a 0.97 correlation cutoff

x
Log File Drive Free Space
Database Size
Database Drive Free Space

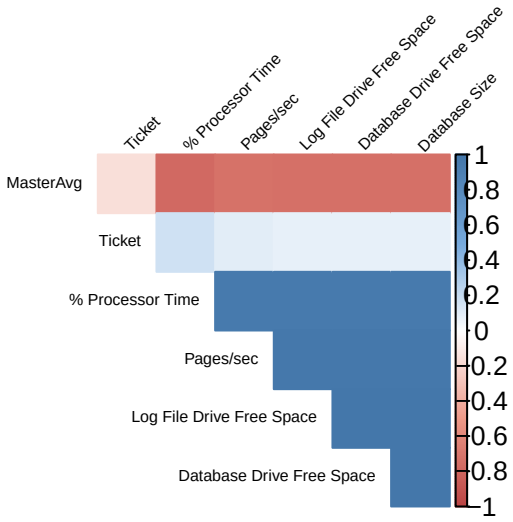
The final dataset was reduced from 21 columns to 12 columns. The visualizations below show the breakdown of the remaining variables and the actual ticket where 1 = ticket and 0 = No ticket. Based on feature correlation it is recommended to remove these additional columns at a .97 cutoff:



The correlation plot confirms that the 3 columns in question are highly correlated and very likely would not

provide any additional value by including:

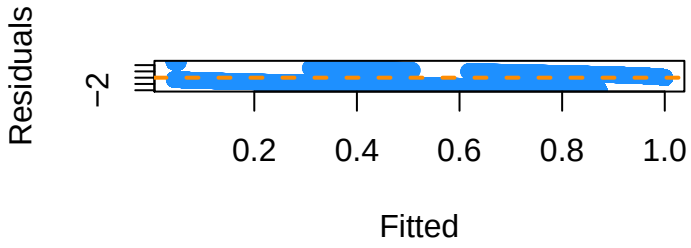
Log File Drive Free Space Database Size Database Drive Free Space

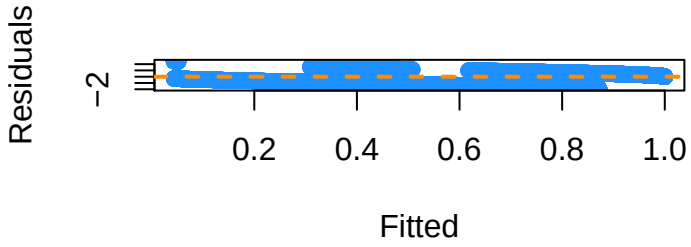


Column Selection

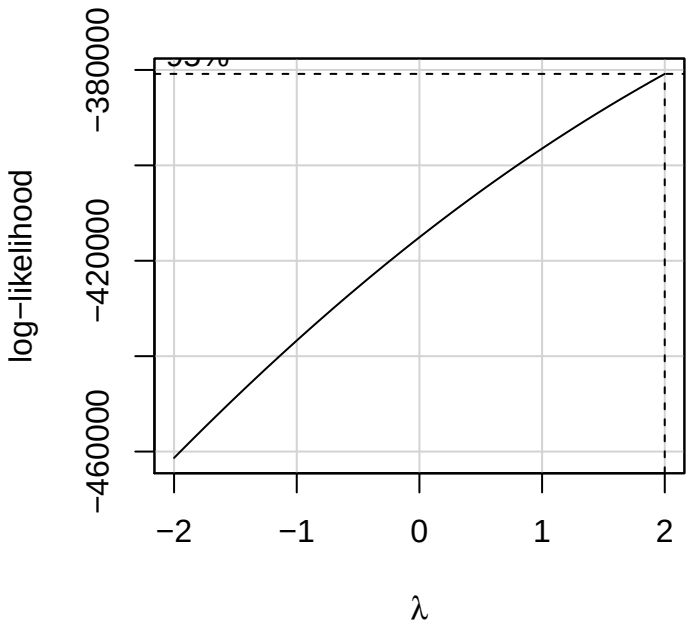
We begin by testing a general linear model and compare to a step GLM. The graphs shows there is no difference when comparing all numerical models to Ticket using GLM or the reduced columns based on a step GLM models. This tells us additional columns can be eliminated as some columns are still highly correlated and would provide similar statistical results.

```
## Start:  AIC=38073
## as.factor(Ticket) ~ (DateTime + ManagedEntityGuid + AlertName +
##     Severity + ServerName + `% Processor Time` + `Database Drive Free Space` +
##     `Database Size` + `Log File Drive Free Space` + `Pages/sec` +
##     MasterAvg) - DateTime - ManagedEntityGuid - AlertName - Severity -
##     ServerName
##
##
## Step:  AIC=38073
## as.factor(Ticket) ~ `% Processor Time` + `Database Drive Free Space` +
##     `Database Size` + `Pages/sec` + MasterAvg
##
##
## Step:  AIC=38073
## as.factor(Ticket) ~ `% Processor Time` + `Database Drive Free Space` +
##     `Pages/sec` + MasterAvg
##
##                                Df Deviance   AIC
## <none>                              38063 38073
## - MasterAvg                     1    38117 38125
## - `Database Drive Free Space`   1    38888 38896
## - `Pages/sec`                   1    45428 45436
## - `% Processor Time`            1    52517 52525
```





We use boxcox function to determine some possible lamda values to transform dependent and independent variables. The boxcox functions results gives 2 as the number of Y dependent variable that should be required for any models. The data frame that follows has the lambda for X independent variables.



X..Processor.Time	Database.Drive.Free.Space	Database.Size	Log.File.Drive.Free.Space	Pages.sec	MasterAvg
-0.4	0	-0.2	0	-1.4	NA

Regsubset and LDA help us narrow down significant columns further. The BIC summary estimate that 3 out of the 6 remaining numerical columns listed below are significant and should be used in modeling.

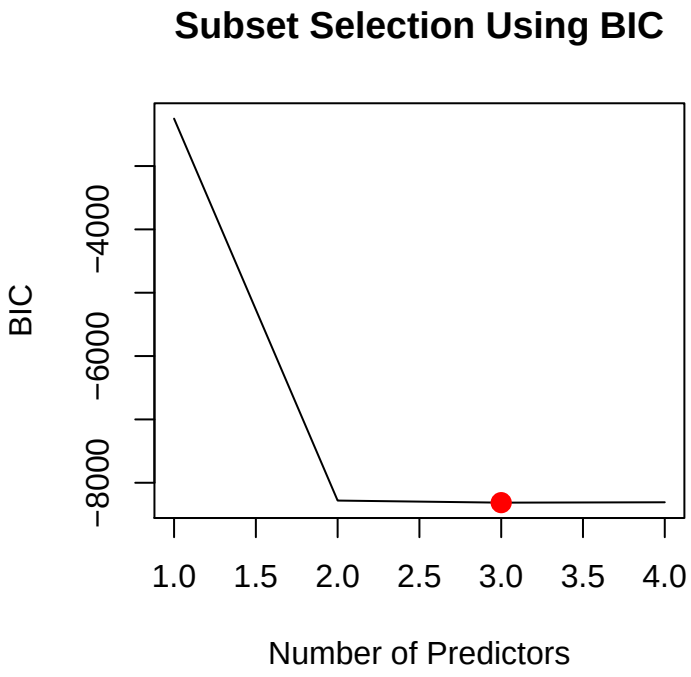
```
## Reordering variables and trying again:
```

Table 3: RegSubsets Results

	'% Processor Time'	'Database Drive Free Space'	'Database Size'	'Log File Drive Free Space'	'Pages/sec'
1 (1)	*				
2 (1)	*	*			
3 (1)	*	*			
4 (1)	*	*			*

Table 4: LDA Results

	LD1
'% Processor Time'	0.22
'Database Drive Free Space'	0.00
'Database Size'	0.00
'Log File Drive Free Space'	0.00
'Pages/sec'	0.00
MasterAvg	-191.28



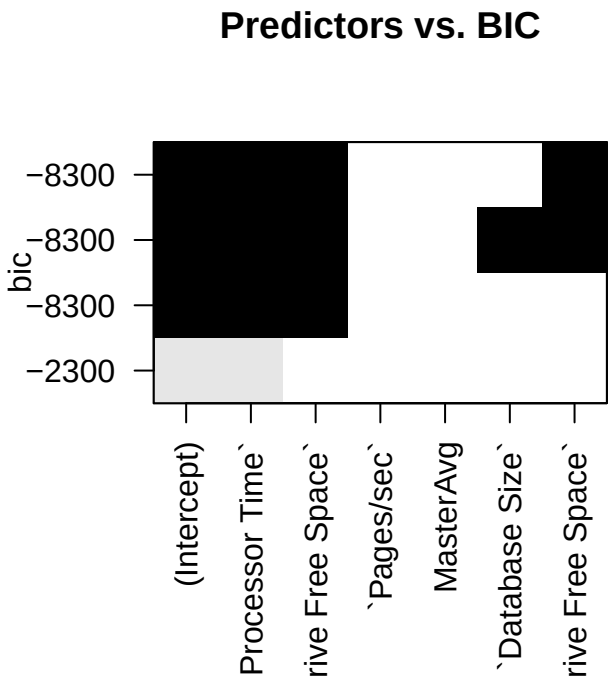


Table 5: MODELS

	RMSE	Rsquared	MAE
CTree	0.203253559355874	0.610870643822876	0.081758073050308
Ridge Regression	0.308525692192921	0.103346088802997	0.208654345879575
Linear Regression	0.308525692192923	0.103346088803007	0.208654345879425
Lasso	0.308637676355344	0.103249457281396	0.206757361635420
SVM	1825127362705.334228515625000	0.000001555195199	271461327403.643035888671875

Modeling Testing

In the model testing we will use 5 model below for comparison and select the best model for the dataset to compare with keras neural network model. Each model test includes the boxcox parameter as a pre processing option. We split the training and testing set by 70/30. :

Linear Regression Ridge Regression Lasso CTree SVM

- **LINEAR- Linear modeling is used for explaining or modeling the relationship between a single variable Y called the response, outcome, output or dependent variable, and one or more predictor, input, independent or explanatory variables.**
- **RIDGE- Ridge regression is a shrinkage techniques that tries to minimize RSS. It uses delta tuning parameters as a shrinkage penalty , which is small when beta is close to zero.**
- **Lasso fixes some shortfalls of Ridge by using a penalty that uses an absolute value of beta to allow variable to be exactly zero, which in turn allow Lasso to perform variable selection. When delta is zero Lasso gives least square fit, when delta is large Lasso gives the null model where all coefficient estimates equal zero. Lasso also tries to find the set of coefficient estimates the lead to the smallest RSS.**
- **CTREE - Decision trees take the shape of a graph that illustrates possible outcomes of different decisions based on a variety of parameters.**
- **SVM - SVM is a supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis**

We will use RMSE, Rsquared, MAE, AUC, ACCURACY, TPR and FPR to select the best model of the 5. We are using the BoxCox prePRocOptions to integrate it into the model.

- **Rsquared - is the coefficient of how well the values fit compared to the original values. The value from 0 to 1 interpreted as percentages. The higher the value is, the better the model is.**
- **RMSE - Root Mean Squared Error is the error rate by the square root of MSE**
- **MAE - is the difference between the original and predicted values extracted by averaged the absolute difference over the data set**

The higher the Rsquared and lower RMSE /MAE the better the model. We can see CTREE has the best model based on these metrics and Ridge/Linear regression are tied for second.

- **TPR - True positive rate (sensitivity) tries to find the percentage of true positives where predictions were correctly identified**
- **FPR - False positives rate (1 - specificity(True Negative)) tries to find percentage of true positive where prediction is incorrectly identified (Type 1 Errors)**
- **(Specificity and sensitivity are inversely proportional, while TPR and FPR are not.)**

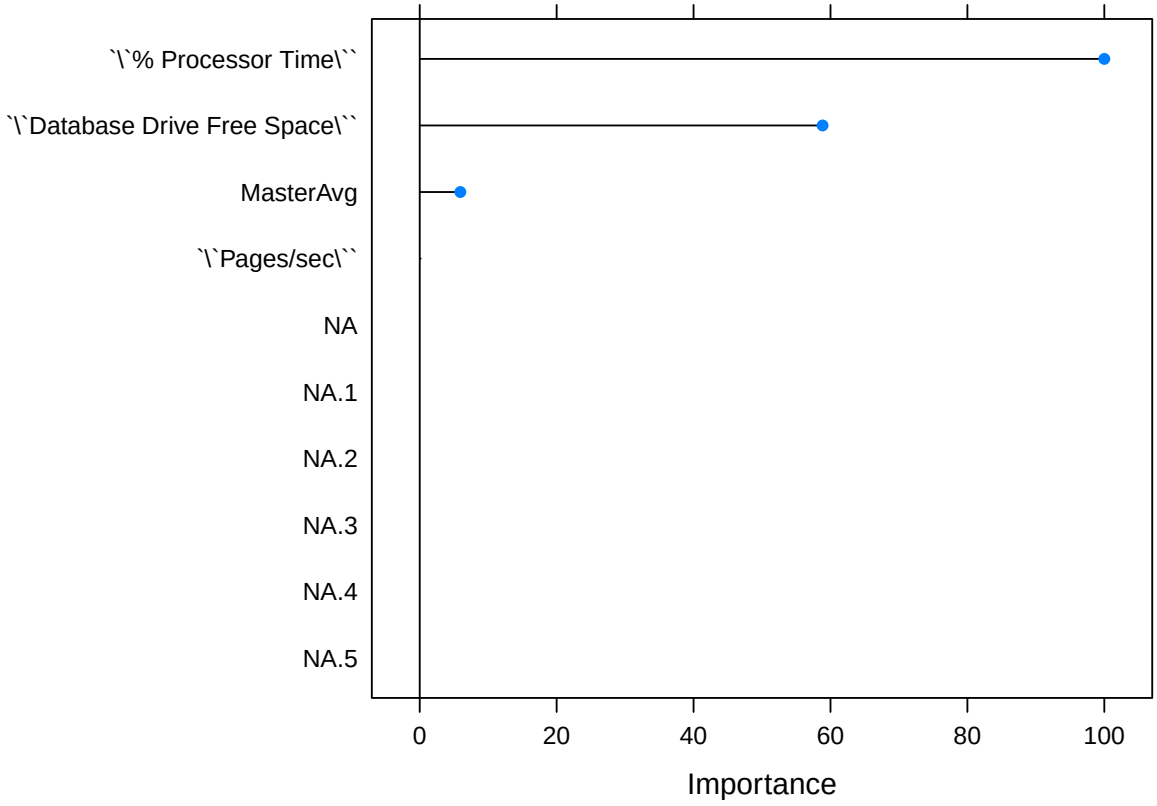
- **Accuracy - Finds the percentage of true positive and true negatives where predictions were correctly identified.**
- **AUC - Area under the curve is the area under the ROC curve when comparing TPR to FPR for several models. The closer the curve is to the upper left corner the better the model classification.**

The AUC/ACC/TPR/FPR comparisons do not differ very much. Ctree has a higher accuracy value although small, it is better than the rest.

ALGO	AUC	ACC	TPR	FPR
LR	0.78	0	0	0
RIDGE	0.78	0	0	0
CT	0.78	0.02	0	0.02
LASSO	0.78	0	0	0
SVM	0.78	0	0	0

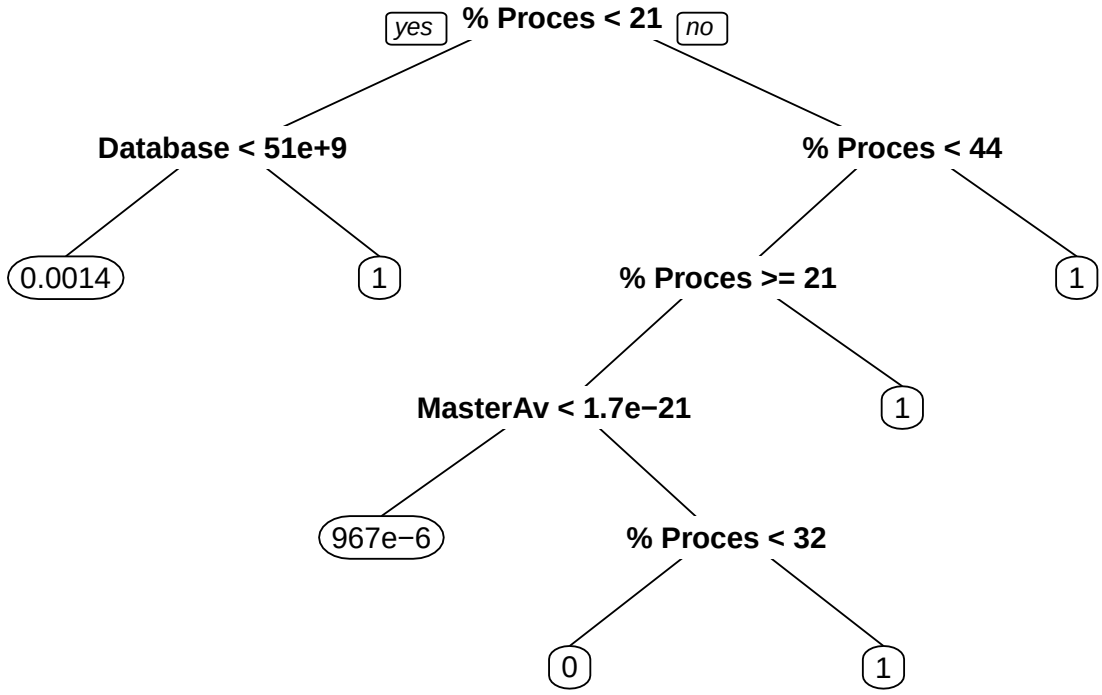
Model Selection

Based on above metrics we will compare CTREE to Keras NN. Below are the top Variables for the Linear Regression. We can see that % Processor Time, Database Drive Free Space and Master Avg variables are important.



	Overall
‘\‘% Processor Time\“	100.00000000000000
‘\‘Database Drive Free Space\“	58.84760509329563
MasterAvg	5.93296572839778
‘\‘Pages/sec\“	0.00000000000000

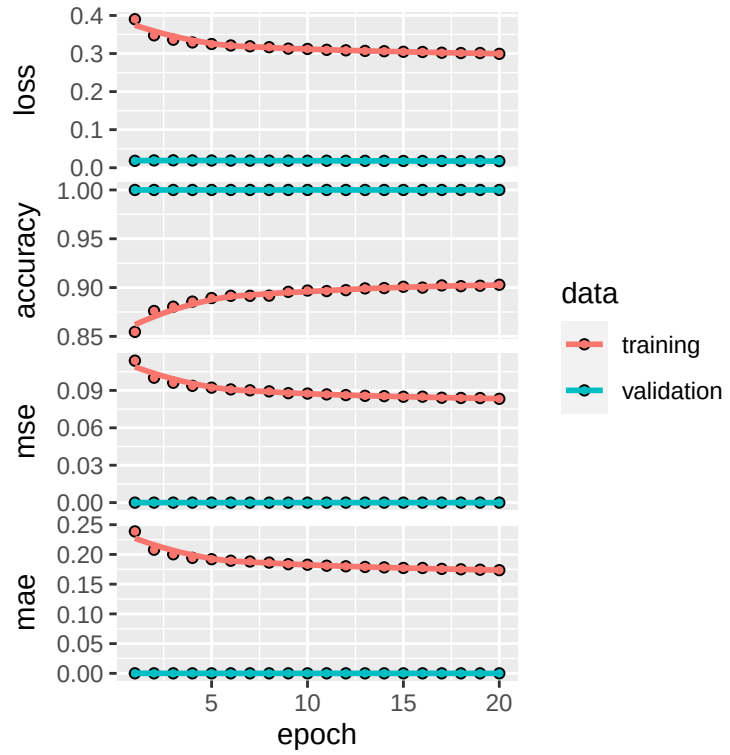
For comparison we also plotted a tree diagram which gave us similar results with the top three predictors also taking the top 3 nodes in the tree.



	x
% Processor Time	5454.947130506595
Database Drive Free Space	2506.022465198165
Database Size	2506.022465198165
Log File Drive Free Space	2506.022465198165
Pages/sec	2193.036602090950
MasterAvg	537.211861960059

The Keras implementation shows accuracy increases and mae/mse decreases at a higher rate than Ctree and is therefor the more accurate and preffered model.

```
## Model: "sequential"
## _________________________________________________________________________________
## Layer (type)                         Output Shape                      Param #
## =================================================================================
## dense_2 (Dense)                      (None, 7)                         49
## _________________________________________________________________________________
## dropout (Dropout)                    (None, 7)                         0
## _________________________________________________________________________________
## dense_1 (Dense)                      (None, 12)                        96
## _________________________________________________________________________________
## dense (Dense)                        (None, 2)                         26
## =================================================================================
## Total params: 171
## Trainable params: 171
## Non-trainable params: 0
## _________________________________________________________________________________
```




```
##              loss          accuracy               mse               mae
## 0.234645918011665 0.928378760814667 0.063294380903244 0.136019736528397
```

```
##           Actual
## Predicted     0     1
##         0  1186   104
##         1  1474 19862
```

	Start	End
mae	0.238725319504738	0.1735209375619888
mse	0.113864496350288	0.0832634046673775
accuracy	0.854610979557037	0.9028633236885071

Problem Resolution

Part of the modeling objective is to predict ticket generation and provide a possible resolution. We update the dataset to include some randomly sampled ad-hoc resolution text. In a real world scenario the separate database /application will be used and merged with predictions.

To simulate, we select data from the original data set based on the indexed rows from the trained dataset and merge it with the predicted dataset from the Keras model. We then generate a random resolution vector based on random text and select each text based on a random sample probabilities and merge that dataset with the original merged data. We further subset the dataset to select only rows where predictions and actual = 1, we group by Alertname, count by resolution and order by resolution count descending. We further mutate by adding sample hypothetical code for automated resolutions.

```
gg<-cbind(nonats2[indx == 2,c(3,5,6,7,8,11)],results[,3:4])

resolutions<-c("restart system services",
               "clear application cache",
               "contact local suport for system reboot",
               "known issue eol application",
               "kill vb process and restart mm service",
               "clear print queue",
               "request more vm memory",
               "request more vm disk space",
               "request for an additional VM CPU core",
               "clear disk space"
               )


gg<-gg%>%mutate(resolution = sample(resolutions, nrow(gg), replace=TRUE,
                                    prob = c(0.1, .2, .15, .05, .1,.05,.1,.1,.1,.05)))

ggp<-subset(gg, pred !=0 & y_test_actual !=0)%>%
        group_by(AlertName)%>%
        count(AlertName,resolution)%>%
        arrange(desc(AlertName,n))%>%
        top_n(2)

ggp<-ggp%>%  mutate(code = case_when
         ((resolution == 'clear application cache') ~ 'clear-cache %servername%',
           (resolution == 'contact local suport for system reboot') ~ 'restart-server %servername%',
           (resolution == 'known issue eol application') ~ 'generate-ticket %servername%',
           (resolution == 'kill vb process and restart mm service') ~ 'stop-process %processname%',
           (resolution == 'clear print queue') ~ 'restart-printqueue %servername%',
           (resolution == 'request more vm memory') ~ 'clear-memcache %servername%',
           (resolution == 'request more vm disk space') ~ 'update-disk %servername%',
           (resolution == 'request for an additional VM CPU core') ~ 'update-cpucore %servername%',
           (resolution == 'clear disk space') ~ 'clear-tempspace %servername'))

write.csv(ggp,"predicted_eval_values_Tickets.csv")

kable(head(ggp))
```

AlertName	resolution	n	c
Total CPU Utilization Percentage is too high	clear application cache	31	c
Total CPU Utilization Percentage is too high	contact local suport for system reboot	29	r
The Schema Master Ping Availability health monitor has failed.	clear application cache	164	c
The Schema Master Ping Availability health monitor has failed.	contact local suport for system reboot	129	r
The Schema Master LDAP Bind Availability health monitor has failed.	clear application cache	164	c
The Schema Master LDAP Bind Availability health monitor has failed.	contact local suport for system reboot	132	r

The final dataset takes the top 2 resolutions for each Alert. This final dataset can be used by a support team to resolve issues either manually step by step or to build automated scripts for automated resolutions. The scripts can be inetgrated into an monitoring systems to take proactive actions based on targeted metrics to resolve issues.

Conclusion

It this paper we have taken some performance data from several development machines to attempt to predict degradation of systems to proactively initiate resolutions. During our data analysis we narrowed down the number of variables. We looked at several model which resulted in selecting decision tree and Keras NN. In the end, Keras showed better performance. In addition, we merged our predicted results dataset with our simulated resolution dataset as a way to initiate automated resolutions.

Although we experienced higher performance with Keras, we tested with a small sample of servers. In addition, these servers were developement servers which would not have he same load as a Production server. Further testing should be done with higher amount of Production serer data , in addition to expanding to other classes of servers. From review of data, these servers were very likely a mix of database and active directory server. It would be good to see how these models and other models would perform on Production servers with heavy loads.

Appendix

Work Cited

[1] Anjitha P, “Web server Performance Prediction using a Deep Recurrent network”,International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056 Volume: 05 Issue: 09 | Sep 2018 www.irjet.net p-ISSN: 2395-0072

[2] T. Vercauteren, P. Aggarwal, X. Wang and T. Li, “Hierarchical Forecasting of Web Server Workload Using Sequential Monte Carlo Training,” in IEEE Transactions on Signal Processing, vol. 55, no. 4, pp. 1286-1297, April 2007, doi: 10.1109/TSP.2006.889401.

[3] Chonggun Kim and H. Kameda, “An algorithm for optimal static load balancing in distributed computer systems,” in IEEE Transactions on Computers, vol. 41, no. 3, pp. 381-384, March 1992, doi: 10.1109/12.127455.

[4] Jin Liang, K. Nahrstedt and Yuanyuan Zhou, “Adaptive multi-resource prediction in distributed resource sharing environment,” IEEE International Symposium on Cluster Computing and the Grid, 2004. CCGrid 2004., Chicago, IL, USA, 2004, pp. 293-300, doi: 10.1109/CCGrid.2004.1336580.

[5] Ziyan Wua, Zhihui Lu a,*, Wei Zhang a, Jie Wua, Shalin Huang b, Patrick C.K. Hung c, " A data-driven approach of performance evaluation for cache server groups in content delivery network",ScienceDirect, J. Parallel Distrib. Comput. 119 (2018) 162–171

[6] Martin Adam1,2, Luca Magnoni2, Martin Piláť3, and Dagmar Adamová1,“Detection of Erratic Behavior in Load Balanced Clusters of Servers Using a Machine Learning Based Method”, EPJ Web of Conferences 214, 08030 (2019) https://doi.org/10.1051/epjconf/201921408030 CHEP 2018

[7] Pavel Barca1*,Bojan Vujanić1,Nemanja Maček2,“MONITORING AND PREDICTING LINUX SERVER PERFORMANCE WITH LINEAR REGRESSION”,SINTEZA 2018 INTERNATIONAL SCIENTIFIC CONFERENCE ON INFORMATION TECHNOLOGY AND DATA RELATED RESEARCH

[8] Zheng Huang,1,2 Jiajun Peng,1 Huijuan Lian,1 Jie Guo,1 and Weidong Qiu1,“Deep Recurrent Model for Server Load and Performance Prediction in Data Center” Wiley,Volume 2017, Article ID 8584252, 10 pages https://doi.org/10.1155/2017/8584252

[9] Per Kreuger Rebecca Steinert Olof Görnerup Daniel Gillblad,“Distributed dynamic load balancing with applications in radio access networks”,Wiley , DOI: 10.1002/nem.2014

[10] Satoru Ohta, “Obtaining the Knowledge of a Server Performance from NonIntrusively Measurable Metrics”,International Journal of Engineering and Technology Innovation, vol. 6, no. 2, 2016, pp. 135 - 151

[11] Shaifu Gupta1 · A. D. Dileep1 · Timothy A. Gonsalves1, “A joint feature selection framework for multivariate resource usage prediction in cloud servers using stability and prediction performance”, The Journal of Supercomputing (2018) 74:6033–6068 https://doi.org/10.1007/s11227-018-2510-7

[12] D. Xikun, W. Huiqiang and L. Hongwu, “A Comprehensive Monitor Model for Self-Healing Systems,” 2010 International Conference on Multimedia Information Networking and Security, Nanjing, Jiangsu, 2010, pp. 751-756, doi: 10.1109/MINES.2010.159.

Other Links

https://www.r-bloggers.com/2018/11/lstm-with-keras-tensorflow/

https://cran.r-project.org/web/packages/simstudy/vignettes/simstudy.html

https://archive.ics.uci.edu/ml/datasets/Wearable+Computing%3A+Classification+of+Body+Postures+and+Movements+%28PUC-Rio%29

https://towardsdatascience.com/7-ways-to-handle-missing-values-in-machine-learning-1a6326adf79e

https://cran.r-project.org/web/packages/imputeTS/vignettes/imputeTS-Time-Series-Missing-Value-Imputation-in-R.pdf

https://riptutorial.com/r/topic/4680/gpu-accelerated-computing

http://www.r-tutor.com/gpu-computing/svm/rpusvm-1

https://xgboost.readthedocs.io/en/latest/tutorials/model.html

https://eigenvector.com/wp-content/uploads/2020/03/Wise_APACT_Nonlinear_Comparison.pdf

http://rwanjohi.rbind.io/2018/04/05/time-series-forecasting-using-lstm-in-r/

https://stackoverflow.com/questions/46485024/how-to-use-boxcoxtrans-function-in-r

https://www.youtube.com/watch?v=iMIWee_PXl8&t=1s

https://www.youtube.com/watch?v=60Bb4H0uRO0&list=PL2HQW7jRz1vHQzJ54CavaAKzBn7AuaWUj&index=48

https://www.youtube.com/watch?v=oueuvXd4O9I

Code

Code used in analysis

```
#rm(list=ls())

knitr::opts_chunk$set(echo = FALSE, message = FALSE, warning = FALSE)
options(scipen=999, digits = 2)

list.of.packages <- c("alluvial","caret","caret","corrplot","corrplot","data.table","dplyr","faraway","
new.packages <- list.of.packages[!(list.of.packages %in% installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages, dependencies=TRUE)


#knitr::opts_chunk$set(echo = TRUE)
require(knitr)
library(ggplot2)
library(tidyr)
library(MASS)
library(psych)
library(kableExtra)
library(dplyr)
library(faraway)
library(gridExtra)
library(reshape2)
library(leaps)
library(pROC)
library(caret)
library(pROC)
library(mlbench)
library(e1071)
library(fpp2)
library(mlr)
library(recommenderlab)
library(jsonlite)
library(stringr)
library(devtools)
library(sparklyr)
library(readtext)
library(simmer)
library(quanteda)
library(tidytext)
library(tm)
library(purrr)
library(topicmodels)
library(RMySQL)
library(plotly)
library(GGally)
library(corrplot)
library(imputeTS)
library(car)
library(rpart)
library(keras)
library(tensorflow)
library(naniar)
```

```
library(reticulate)
setwd(getwd())
d1<- as.data.frame(read.csv('../Datasets/TestQuery.csv', header = TRUE,quote = "\"",sep=','))
cnames<-c('AlertName','AlertDescription','Severity','Priority','DateTime','ManagedEntityGuid')
d2<- as.data.frame(read.csv('../Datasets/TestQuery3.csv',header= FALSE, col.names = cnames, quote = "\"

joineddata<-right_join(d1, d2,  by = c('DateTime','ManagedEntityGuid'))
joineddata<-subset(joineddata,!is.na(SampleValue))%>%
  subset(select=c(-AlertName.x,-Severity.x))%>%
  rename(AlertName = AlertName.y, Severity = Severity.y)%>%
  mutate(ServerName = case_when
         ((ManagedEntityGuid == '2A909362-BE1E-F148-835C-96787AE2775E') ~ 'Server1',
         (ManagedEntityGuid == '6EA459BD-0616-71F6-FC4A-7989A2DCF9FD') ~ 'Server2',
         (ManagedEntityGuid == 'A1A8A538-8B75-B8F5-5F44-7159F192F602') ~ 'Server3',
         (ManagedEntityGuid == 'AF487CE3-8C62-2C7A-F748-9241D73F0403') ~ 'Server4',
          TRUE ~ 'Server5'))%>%
  mutate(Ticket = case_when
         ((Severity == 2) ~ 1,
           TRUE ~ 0))
nm1<-names(d1)
nm2<-names(d2)


nm3<-names(joineddata)
Data_Description<-c('Date stamp of alert','Identity of monitored object','Name of component', 'Performa
def<-as.data.frame(Data_Description,nm3)
kable(def)


glimpse(joineddata)
kable(as.data.frame(table(joineddata$Ticket, joineddata$ServerName)))


#TRANSFORM

sdata<-joineddata%>%subset(select=c(-Objectname, -AlertDescription))%>%
  group_by(DateTime, ServerName, CounterName, AlertName)%>%
  mutate(row_id=1:n()) %>% ungroup() %>%
  spread(CounterName, SampleValue)%>%
  subset(select=c(-row_id,-Priority))

var_stats<- function(df){
  wt<-data.frame(columns=colnames(df))
  wt$na_count <- sapply(df, function(y) sum(is.na(y)))
  wt$neg_count <- sapply(df, function(y) sum(y<0))
  wt$zero_count <- sapply(df, function(y) sum(as.integer(y)==0))
  wt$unique_count <- sapply(df, function(y) sum(n_distinct(y)))
  return(wt)
}

vis_miss(sdata, warn_large_data = F)
gg_miss_upset(sdata)
#kable(var_stats(sdata))
```

```
#Impute NA values by datetime series With Klaman https://cran.r-project.org/web/packages/imputeTS/vigne

nonats<-na_kalman(sdata)
vis_miss(nonats, warn_large_data = F)
#kable(var_stats(nonats))


#Remove highly correllated Ops master* column, replace with avg

corrplot(as.matrix(cor(nonats[c(6:ncol(nonats))]),na.rm=T,method = "shade", use="pairwise.complete.obs"
#kable(round(cor(nonats[c(6:ncol(nonats))]),2))
#ggpairs(nonats[c(6:ncol(nonats))], na.rm = T)



df <- nonats %>% select(starts_with("Op Master"))
nonats<-nonats%>%mutate(MasterAvg=rowMeans(cbind(nonats[names(df)])))
nonats<-nonats%>%select(!c(names(df)))
vis_miss(nonats, warn_large_data = F)
#remove any columns with NA
nonats2<-nonats%>%
  select(where(~!any(is.na(.))))

vis_miss(nonats2, warn_large_data = F)
#corrplot(as.matrix(cor(nonats2[c(6:ncol(nonats2))]),na.rm=T,method = "shade", use="pairwise.complete.o
#kable(round(cor(nonats2[c(6:ncol(nonats2))]),2),na.rm=T)
ggpairs(nonats2[c(6:ncol(nonats2))], na.rm = T)



kable(names(nonats2), cap = "Final Dataset 12 columnns")

kable(names(nonats2[7:ncol(nonats2)])[findCorrelation(cor(nonats2[7:ncol(nonats2)]), cutoff = .97)],
        cap = "Variables recommended for removal at a 0.97 correlation cutoff")

log(nonats2[c(6:ncol(nonats2))]) %>%
  gather(-Ticket, key = "var", value = "val") %>%
  ggplot(aes(x = val, fill=as.factor(Ticket))) +
  geom_histogram(bins=10, alpha=1) +
  facet_wrap(~ var, scales = "free") +
  scale_fill_manual("Ticket",
                    values = c('#d73027','#fc8d59','#fee090',
                               '#e0f3f8','#91bfdb','#4575b4')) +
  xlab("") +
  ylab("") +
  theme(panel.background = element_blank(), legend.position="top")


featurePlot(log(nonats2[7:ncol(nonats2)]),nonats2$Ticket)
```

```r
cor.plt <- cor(nonats2[c(6:ncol(nonats2))], use = "pairwise.complete.obs", method = "pearson")
col <- colorRampPalette(c("#BB4444", "#EE9988", "#FFFFFF", "#77AADD", "#4477AA"))
corrplot(cor.plt, method="color", col=col(200),
         type="upper", order="hclust",
         tl.col="black", tl.srt=45, tl.cex=0.5,
         diag=FALSE
         )


glmod<-glm(Ticket~.-DateTime-ManagedEntityGuid-AlertName-Severity-ServerName,nonats2,family = binomial(

smod<-step(glm(as.factor(Ticket)~.-DateTime-ManagedEntityGuid-AlertName-Severity-ServerName,nonats2,fam

par(mfrow = c(2,1))
plot(fitted(glmod), resid(glmod), col = "dodgerblue",
     pch = 20, cex = 1.5, xlab = "Fitted", ylab = "Residuals")
abline(h = 0, lty = 2, col = "darkorange", lwd = 2)

plot(fitted(smod), resid(smod), col = "dodgerblue",
     pch = 20, cex = 1.5, xlab = "Fitted", ylab = "Residuals")
abline(h = 0, lty = 2, col = "darkorange", lwd = 2)


boxCox(glmod, family="yjPower", plotit = TRUE)

nonats_bct <- apply(nonats2[7:ncol(nonats2)], 2, BoxCoxTrans)
kable(as.data.frame(lapply(nonats_bct, function(x) x$lambda)))



regfit.full <- regsubsets(Ticket ~ .-DateTime-ManagedEntityGuid-AlertName-Severity-ServerName, data=nona
reg.summary <- summary(regfit.full)
plot(reg.summary$bic, xlab="Number of Predictors", ylab="BIC", type="l", main="Subset Selection Using B

BIC_num <- which.min(reg.summary$bic)
points(BIC_num, reg.summary$bic[BIC_num], col="red", cex=2, pch=20)
reg.summary <- summary(regfit.full)
plot(regfit.full, scale="bic", main="Predictors vs. BIC")

kable(reg.summary$outmat, caption = "RegSubsets Results")

llda<-lda(Ticket~.-DateTime-ManagedEntityGuid-AlertName-Severity-ServerName,nonats2)
kable(llda$scaling,caption = "LDA Results")


#Data 624 technical report
#Partition Data
nn<-nonats2[c(6:ncol(nonats2))]
set.seed(123)
trainidx<-sample(nrow(nn),round(0.7*nrow(nn)),replace=F)
traindata<-nn[trainidx,]
testdata<-nn[-trainidx,]
```

```
require(caret)
set.seed(555)
trctrl<- trainControl(method="repeatedcv", number=3, repeats=2, preProcOptions  = "BoxCox")

##Linear Regression
linreg <- caret::train(Ticket~., data=traindata, method="lm",
                trControl=trctrl)
linPred <- predict(linreg, newdata = testdata)
m1<-data.frame(postResample(pred = linPred, obs = testdata$Ticket)) #0.1414880 0.3775156

confusionMatrix<-table(linPred,testdata$Ticket,dnn=c('Predicted','Actual'))
lr.acc<-round(sum(diag(confusionMatrix))/sum(confusionMatrix),2)
lr.miss<-round((confusionMatrix[1,2]+confusionMatrix[1,2])/sum(confusionMatrix),2)
lr.tpr<-round(sum(confusionMatrix[1,2])/sum(confusionMatrix[,2]),2)
lr.fpr<-round(sum(confusionMatrix[1,1])/sum(confusionMatrix[,2]),2)
lrr<-roc(Ticket~linPred,data=testdata)
lr.auc<-round(lrr$auc[1],2)

##Ridge Regression
ridge <- caret::train(Ticket~., data=traindata, method="ridge",
                trControl=trctrl)
ridgePred <- predict(ridge, newdata = testdata)
m2<-data.frame(postResample(pred = ridgePred, obs = testdata$Ticket)) #0.1414837 0.3775762

confusionMatrix<-table(ridgePred,testdata$Ticket,dnn=c('Predicted','Actual'))
rd.acc<-round(sum(diag(confusionMatrix))/sum(confusionMatrix),2)
rd.miss<-round((confusionMatrix[1,2]+confusionMatrix[1,2])/sum(confusionMatrix),2)
rd.tpr<-round(sum(confusionMatrix[1,2])/sum(confusionMatrix[,2]),2)
rd.fpr<-round(sum(confusionMatrix[1,1])/sum(confusionMatrix[,2]),2)
lrd<-roc(Ticket~linPred,data=testdata)
lrd.auc<-round(lrd$auc[1],2)


##Lasso Regression
lasso <- caret::train(Ticket~., data=traindata, method="lasso",
                trControl=trctrl)
lassoPred <- predict(lasso, newdata = testdata)
m3<-data.frame(postResample(pred = lassoPred, obs = testdata$Ticket)) #0.1418941 0.3762947

confusionMatrix<-table(lassoPred,testdata$Ticket,dnn=c('Predicted','Actual'))
la.acc<-round(sum(diag(confusionMatrix))/sum(confusionMatrix),2)
la.miss<-round((confusionMatrix[1,2]+confusionMatrix[1,2])/sum(confusionMatrix),2)
la.tpr<-round(sum(confusionMatrix[1,2])/sum(confusionMatrix[,2]),2)
la.fpr<-round(sum(confusionMatrix[1,1])/sum(confusionMatrix[,2]),2)
lla<-roc(Ticket~linPred,data=testdata)
lla.auc<-round(lla$auc[1],2)

##CTree
ctre <- caret::train(Ticket~., data=traindata, method="ctree2",
                trControl=trctrl)
ctrePred <- predict(ctre, newdata = testdata)
m4<-data.frame(postResample(pred = ctrePred, obs = testdata$Ticket)) #0.1519582 0.2804008
```

```
confusionMatrix<-table(ctrePred,testdata$Ticket,dnn=c('Predicted','Actual'))
ct.acc<-round(sum(diag(confusionMatrix))/sum(confusionMatrix),2)
ct.miss<-round((confusionMatrix[1,2]+confusionMatrix[1,2])/sum(confusionMatrix),2)
ct.tpr<-round(sum(confusionMatrix[1,2])/sum(confusionMatrix[,2]),2)
ct.fpr<-round(sum(confusionMatrix[1,1])/sum(confusionMatrix[,2]),2)
lct<-roc(Ticket~linPred,data=testdata)
lct.auc<-round(lrr$auc[1],2)

##SVM (Radial Kernel)
svmGrid <-  expand.grid(C = c(1,1000))
svmFit <- caret::train(Ticket~., data = traindata,
                  #type='eps-regression',
                  method = 'svmRadialCost',
                  trControl = trctrl,
                  tuneGrid = svmGrid)
svmPred <- predict(svmFit, newdata = testdata)
m5<-data.frame(postResample(pred = svmPred, obs = testdata$Ticket)) #0.13136218 0.48751241

confusionMatrix<-table(svmPred,testdata$Ticket,dnn=c('Predicted','Actual'))
sv.acc<-round(sum(diag(confusionMatrix))/sum(confusionMatrix),2)
sv.miss<-round((confusionMatrix[1,2]+confusionMatrix[1,2])/sum(confusionMatrix),2)
sv.tpr<-round(sum(confusionMatrix[1,2])/sum(confusionMatrix[,2]),2)
sv.fpr<-round(sum(confusionMatrix[1,1])/sum(confusionMatrix[,2]),2)
lsv<-roc(Ticket~linPred,data=testdata)
lsv.auc<-round(lrr$auc[1],2)


df<-data.frame(rbind(m1[,1],m2[,1],m3[,1],m4[,1],m5[,1]))

rownames(df)<-c("Linear Regression","Ridge Regression","Lasso","CTree","SVM")
colnames(df)<-c("RMSE","Rsquared","MAE")
df <- df[order(df$RMSE),]
options(digits = 15)
kable(df, cap="MODELS")

finaldf<-data.frame(ALGO=c('LR','RIDGE','CT','LASSO','SVM'),
                    AUC=c(lr.auc,lrd.auc,lct.auc,lla.auc,lsv.auc),
                    ACC=c(lr.acc,rd.acc,ct.acc,la.acc,sv.acc),
                    TPR=c(lr.tpr,rd.tpr,ct.tpr,la.tpr,sv.tpr),
                    FPR=c(lr.fpr,rd.fpr,ct.fpr,la.fpr,sv.fpr))



#Variable Importance Ranking (Liner Regression)
rfImp <- varImp(linreg, scale = TRUE, conditional= TRUE)
plot(rfImp, top=10, scales = list(y = list(cex = 0.8)))



rfImp2 <- rfImp$importance[order(-rfImp$importance$Overall), , drop=FALSE]
head(rfImp2, 10)

require(rpart.plot)
```

```r
tree <- rpart(Ticket~., data=traindata)
prp(tree)
kable(tree$variable.importance)


eval_p2 <- predict(ctre, newdata = testdata[-1])

PHMut <- mutate(testdata, predProb = predict(ctre ,testdata[-1],type = "raw"))
grpPH <- group_by(PHMut, cut(eval_p2, breaks = unique(quantile(eval_p2, (0:25)/26, na.rm=TRUE))))

#hosmer-lemeshow stat
hlDf <- summarise(grpPH, y= sum(Ticket), pPred=mean(predProb), count = n())
hlDf <- mutate(hlDf, se.fit=sqrt(pPred * (1-(pPred)/count)))
ggplot(hlDf,aes(x=pPred,y=y/count,ymin=y/count-2*se.fit,ymax=y/count+2*se.fit)) +
    geom_point()+geom_linerange(color=grey(0.75))+geom_abline(intercept=0,slope=1) +
    xlab("Predicted Probability") +
    ylab("Observed Proportion")


#Introducing Keras for deep learning in R (https://www.youtube.com/watch?v=6OBb4HOuROQ&list=PL2HQW7jRz1

nn<-nonats2[c(6:ncol(nonats2))]


df<-nn# Convert the data.frame to a matrix
data <- as.matrix(df)
# Remove variable names
dimnames(data) <- NULL

# Split for train and test data
set.seed(123)
indx <- sample(2,
               nrow(data),
               replace = TRUE,
               prob = c(0.7, 0.3)) # Makes index with values 1 and 2


x_train <- data[indx == 1, 2:7] # Take rows with index = 1
x_test <- data[indx == 2, 2:7]


#A separate computer variable is created to hold the ground-truth (actual) feature values of the test s

#y_test_actual <- data[indx == 2, 22]
y_test_actual <- data[indx == 2, 1]

#The feature variables of the training and test sets can be one-hot-encoded using the Keras function to

# Using similar indices to correspond to the training and test set
y_train <-to_categorical(data[indx == 1, 1])
y_test <- to_categorical(data[indx == 2, 1])

#The code chunk below calculates the mean and the standard deviation of the training set and then norma
```

```r
mean_train <- apply(x_train,
                    2,
                    mean)
std_train <- apply(x_train,
                   2,
                   sd)
x_train <- scale(x_train,
                 center = mean_train,
                 scale = std_train)
x_test <- scale(x_test,
                center = mean_train,
                scale = std_train)


#The model is created uses L2 regularization to combat overfitting. The rectified linear unit, relu, ac

# Creating the model
model <- keras_model_sequential()
model %>%
  layer_dense(units = 7,
              kernel_regularizer = regularizer_l2(0.001),
              activation = "relu",
              input_shape = c(6)) %>%
  layer_dropout(rate = 0.2) %>%
  layer_dense(units = 12,
              kernel_regularizer = regularizer_l2(0.001),
              activation = "relu") %>%
  layer_dense(units = 2,
              activation = "softmax")
summary(model)


#Compiling the model Before fitting, the model requires compilation. The loss function, optimizer, and

model %>% compile(loss = "categorical_crossentropy",
                  optimizer = "adam",
                  metrics = c("accuracy","mse", "mae"))



#Fitting the data The training set is used to fit the compiled model.

history <- model %>%
  fit(x_train,
      y_train,
      epoch = 20,
        batch_size = 64,
      validation_split = 0.2)

#A simple plot can be created to show the loss and the accuracy over the epochs.

plot(history)
```

```
model%>%
  evaluate(x_train,
           y_train)

pred<-model%>%
  predict_classes(x_test)

table(Predicted = pred,
      Actual = y_test_actual)

prob<-model%>%
  predict_proba(x_test)

results<-cbind(prob,
      pred,
      y_test_actual)

#Dataframe with metrics
df2 <- data.frame(matrix(ncol = 2, nrow = 3))
rownames(df2)<-c('mae','mse','accuracy')
colnames(df2)<-c('Start','End')

df2[,1]<-rbind(
  history$metrics$mae[1],
  history$metrics$mse[1],
  history$metrics$accuracy[1])

df2[,2]<-rbind(
  history$metrics$mae[length(history$metrics$mae)],
  history$metrics$mse[length(history$metrics$mse)],
  history$metrics$accuracy[length(history$metrics$accuracy)])

kable(df2)


gg<-cbind(nonats2[indx == 2,c(3,5,6,7,8,11)],results[,3:4])

resolutions<-c("restart system services",
               "clear application cache",
               "contact local suport for system reboot",
               "known issue eol application",
               "kill vb process and restart mm service",
               "clear print queue",
               "request more vm memory",
               "request more vm disk space",
               "request for an additional VM CPU core",
               "clear disk space"
               )


gg<-gg%>%mutate(resolution = sample(resolutions, nrow(gg), replace=TRUE,
                                    prob = c(0.1, .2, .15, .05, .1,.05,.1,.1,.1,.05)))
```

```
ggp<-subset(gg, pred !=0 & y_test_actual !=0)%>%
        group_by(AlertName)%>%
        count(AlertName,resolution)%>%
        arrange(desc(AlertName,n))%>%
        top_n(2)

ggp<-ggp%>%  mutate(code = case_when
         ((resolution == 'clear application cache') ~ 'clear-cache %servername%',
           (resolution == 'contact local suport for system reboot') ~ 'restart-server %servername%',
           (resolution == 'known issue eol application') ~ 'generate-ticket %servername%',
           (resolution == 'kill vb process and restart mm service') ~ 'stop-process %processname%',
           (resolution == 'clear print queue') ~ 'restart-printqueue %servername%',
           (resolution == 'request more vm memory') ~ 'clear-memcache %servername%',
           (resolution == 'request more vm disk space') ~ 'update-disk %servername%',
           (resolution == 'request for an additional VM CPU core') ~ 'update-cpucore %servername%',
           (resolution == 'clear disk space') ~ 'clear-tempspace %servername'))

write.csv(ggp,"predicted_eval_values_Tickets.csv")

kable(head(ggp))
```