Team DAY Project

Dustin Bracy, Adam Ruthford, Yang Zhang

1/23/2020

library(tidyverse)

## -- Attaching packages ----------------------------------------------------------------------------------- tidyverse 1.2.1 --

## v ggplot2 3.2.1 v purrr 0.3.2  
## v tibble 2.1.3 v dplyr 0.8.3  
## v tidyr 0.8.3 v stringr 1.4.0  
## v readr 1.3.1 v forcats 0.4.0

## -- Conflicts -------------------------------------------------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(data.table) #for %like% function

##   
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':  
##   
## between, first, last

## The following object is masked from 'package:purrr':  
##   
## transpose

library(xml2)  
library(purrr)  
library(lubridate)

##   
## Attaching package: 'lubridate'

## The following objects are masked from 'package:data.table':  
##   
## hour, isoweek, mday, minute, month, quarter, second, wday,  
## week, yday, year

## The following object is masked from 'package:base':  
##   
## date

library(tictoc)  
library(kableExtra)

##   
## Attaching package: 'kableExtra'

## The following object is masked from 'package:dplyr':  
##   
## group\_rows

library(car)

## Loading required package: carData

##   
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':  
##   
## recode

## The following object is masked from 'package:purrr':  
##   
## some

tic('total runtime')  
load("./data/reports.RData")  
load("./data/customGroups.RData")  
load("./data/currencyCode.RData")

## Objective 2

We ran a two-way ANOVA on the data to satisfy the second objective. The business requirements drove the selection of the explanatory variables. The business wants to know what time of day and on which server new reports should be added based upon current report run times as expressed by the ReportDeliveryTime response variable.

###Explanatory and Response variable creation

The two explanatory variables require some explanation. The “server” variable required us to remove the data associated with testing and development servers. The testing and development report run times have very different characteristics than production data. Thus, all data associated with servers “SQLODR1” and “SQLODR5” were removed.  
  
“HourBinned” variable indicates the time of day the report ran. Several things should be noted about it. A maintenance window exists every night between the hours of midnight and six AM. Only a few reports run during this maintenance window. Likewise, relatively few reports run during the late evening hours between eight PM and midnight. We binned these two time periods separately. The remaining time during the days was binned in two-hour blocks. The summary tables below give counts and means of the various bins.

reports$HourBinned <- case\_when(reports$HourOfDay %in% c("06","07") ~ "0607",  
 reports$HourOfDay %in% c("08","09") ~ "0809",  
 reports$HourOfDay %in% c("10","11")~ "1011",  
 reports$HourOfDay %in% c("12","13")~ "1213",  
 reports$HourOfDay %in% c("14","15")~ "1415",  
 reports$HourOfDay %in% c("16","17")~ "1617",  
 reports$HourOfDay %in% c("18","19")~ "1819",  
 reports$HourOfDay %in% c("20","21","22","23")~ "2023",  
 reports$HourOfDay %in% c("00","01","02","03","04","05")~ "0005",  
 TRUE~"OTHER")  
  
mysummary<-function(x){  
 result<-c(length(x),mean(x),sd(x),sd(x)/length(x))  
 names(result)<-c("N","Mean","SD","SE")  
 return(result)  
}  
  
servestats <-aggregate(ReportDeliveryTime~Server,data=reports,mysummary)  
servestats.df <- data.frame(servestats[,1:1],servestats[,-(1)])  
servestats.df

## servestats...1.1. N Mean SD SE  
## 1 SQLODR2 92385 2256.028 6880.241 0.07447358  
## 2 SQLODR3 76352 1968.877 6369.361 0.08342101  
## 3 SQLODR6 95400 2161.752 6767.481 0.07093795

The table above shows that server “SQLODR3” runs fewer reports than the other two servers and has differences in mean report run time and standard deviation.

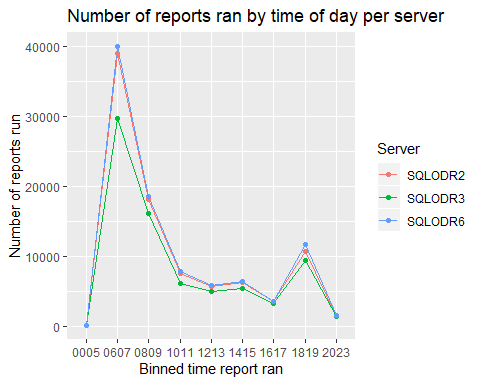
hourstats <-aggregate(ReportDeliveryTime~HourBinned,data=reports,mysummary)  
hourstats.df <- data.frame(hourstats[,1:1],hourstats[,-(1)])  
hourstats.df

## hourstats...1.1. N Mean SD SE  
## 1 0005 384 69285.472 13610.750 35.44466176  
## 2 0607 108726 2021.303 4943.980 0.04547192  
## 3 0809 52825 1589.290 4707.032 0.08910615  
## 4 1011 21445 3045.266 7553.906 0.35224555  
## 5 1213 16435 2525.486 9787.416 0.59552270  
## 6 1415 18022 1483.042 5885.803 0.32658992  
## 7 1617 10284 3169.675 8681.578 0.84418305  
## 8 1819 31846 1358.025 5025.376 0.15780243  
## 9 2023 4170 6059.903 16035.427 3.84542610

sumstats<-aggregate(ReportDeliveryTime~HourBinned:Server,data=reports,mysummary)  
sumstats<-cbind(sumstats[,1:2],sumstats[,-(1:2)])  
#head(sumstats, 30)

Looking at the data above for the BinnedHour variable the late evening hours and the overnight maintenance window bins display very different standard deviations than the rest of the bins. Also, of note the sample count size differs for those two bins. Finally reports should not be added to the overnight bin that includes the maintenance window.

ggplot(mapping=aes(x=HourBinned,y=N,color=Server,group=Server),data=sumstats)+  
 geom\_line()+  
 geom\_point()+ylab("Number of reports run") + xlab("Binned time report ran")+  
 ggtitle("Number of reports ran by time of day per server")

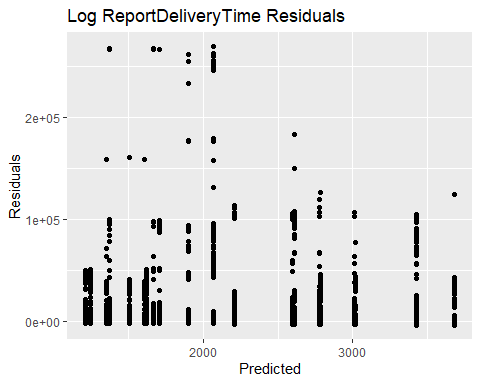
 The graph show that each server tracks the others in terms of reports run at a particular hour. A simple interpretation of the results benefits the business directly. The first analysis of the data included an unlogged response variable. The bins for overnight and late evening (“0005”,”2023”)are excluded from the analysis.

#Remove early morning late evening  
  
reports.fltr <- reports %>% filter(!HourBinned == "0005" & !HourBinned == "2023")  
  
model.fit.HrBnSrv <- aov(ReportDeliveryTime~Server+HourBinned+HourBinned:Server, data = reports.fltr)  
anova.disp <- Anova(model.fit.HrBnSrv,type=3,singular.ok=FALSE)  
  
anova.disp[4]

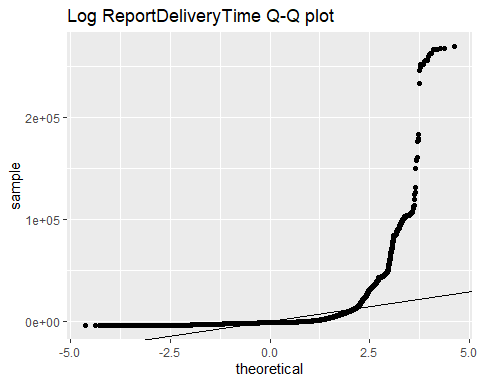
## Pr(>F)   
## (Intercept) < 2.2e-16 \*\*\*  
## Server 9.744e-05 \*\*\*  
## HourBinned < 2.2e-16 \*\*\*  
## Server:HourBinned < 2.2e-16 \*\*\*  
## Residuals   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

The type 3 ANOVA table indicates that both terms as well as the interaction term are statistically significant with very low p-values.

myfits<-data.frame(fitted.values=model.fit.HrBnSrv$fitted.values,residuals=model.fit.HrBnSrv$residuals)  
  
#Residual vs Fitted  
  
plot1<-ggplot(myfits,aes(x=fitted.values,y=residuals))+ylab("Residuals")+  
 xlab("Predicted")+geom\_point()+ggtitle("Log ReportDeliveryTime Residuals")  
  
#QQ plot of residuals #Note the diagonal abline is only good for qqplots of normal data.  
plot2<-ggplot(myfits,aes(sample=residuals))+  
 stat\_qq()+geom\_abline(intercept=mean(myfits$residuals), slope = sd(myfits$residuals))+  
 ggtitle("Log ReportDeliveryTime Q-Q plot")  
  
#Histogram of residuals  
plot3<-ggplot(myfits, aes(x=residuals)) +   
 geom\_histogram(aes(y=..density..),binwidth=1,color="black", fill="gray")+  
 geom\_density(alpha=.1, fill="red")+  
 ggtitle("Log ReportDeliveryTime Histogram")  
  
plot1



plot2



# plot3 Don't run this one takes too long and we are not going to use it anyway

The residual plot appears to demonstrate a cone shape with residuals beneath 2000 being noticeably more spread out than those above 3000. The Q-Q plot displays dramatic differences from normality. A log transform of the response variable is necessary.

reports.fltr$ReportDeliveryTimeLog <- log(reports.fltr$ReportDeliveryTime)  
  
model.fit.HrBnSrvlg <- aov(ReportDeliveryTimeLog~Server+HourBinned+HourBinned:Server, data = reports.fltr)  
anova.fltr.disp <- Anova(model.fit.HrBnSrvlg,type=3,singular.ok=FALSE)  
  
anova.fltr.disp[4]

## Pr(>F)   
## (Intercept) < 2.2e-16 \*\*\*  
## Server 7.604e-06 \*\*\*  
## HourBinned < 2.2e-16 \*\*\*  
## Server:HourBinned < 2.2e-16 \*\*\*  
## Residuals   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

The type III ANOVA table again shows that both the interaction term as well as the individual terms are significant.

mysummary<-function(x){  
 result<-c(length(x),mean(x),sd(x),sd(x)/length(x))  
 names(result)<-c("N","Mean","SD","SE")  
 return(result)  
}  
  
servestats.fltr <-aggregate(ReportDeliveryTimeLog~Server,data=reports.fltr,mysummary)  
servestats.fltr.df <- data.frame(servestats.fltr[,1:1],servestats.fltr[,-(1)])  
servestats.fltr.df

## servestats.fltr...1.1. N Mean SD SE  
## 1 SQLODR2 90870 6.150302 2.098573 2.309423e-05  
## 2 SQLODR3 74943 5.976943 2.099649 2.801661e-05  
## 3 SQLODR6 93770 6.072306 2.134849 2.276687e-05

The mean and standard deviation of the logged ReportDeliveryTime track extremely close now with respect to the server variable.

hourstats.fltr <-aggregate(ReportDeliveryTimeLog~HourBinned,data=reports.fltr,mysummary)  
hourstats.fltr.df <- data.frame(hourstats.fltr[,1:1],hourstats.fltr[,-(1)])  
hourstats.fltr.df

## hourstats.fltr...1.1. N Mean SD SE  
## 1 0607 108726 7.134787 1.200840 1.104465e-05  
## 2 0809 52825 5.644595 2.022867 3.829374e-05  
## 3 1011 21445 5.060337 2.841504 1.325019e-04  
## 4 1213 16435 4.503144 2.569200 1.563249e-04  
## 5 1415 18022 4.776321 2.349070 1.303446e-04  
## 6 1617 10284 5.155736 2.742066 2.666342e-04  
## 7 1819 31846 5.673145 1.672099 5.250578e-05

sumstats.log.fltr<-aggregate(ReportDeliveryTimeLog~HourBinned:Server,data=reports.fltr,mysummary)  
sumstats.log.fltr<-cbind(sumstats.log.fltr[,1:2],sumstats.log.fltr[,-(1:2)])  
#sumstats.log.fltr

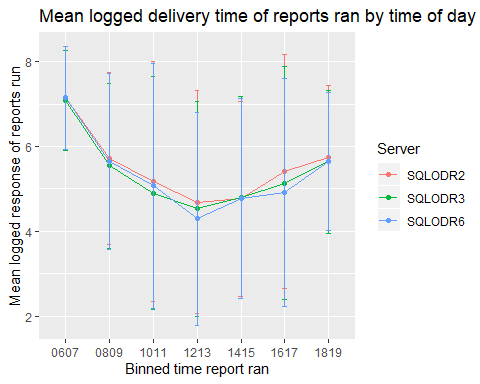
The HourBinned variable does display some differences in standard deviation. This issue is examined below. Please note that the over night and late evening hours were dropped from the model.

The difference in standard deviation between the various groups is concerning. On-line research yielded the following article https://link.springer.com/article/10.3758/s13428-017-0918-2?shared-article-renderer#Tab6 . The article indicates that a useful measure of unequal variance is the Variance ratio. This ratio is simply the largest group variance divided by the smallest group variance. Looking at the summary statistics table for hours binned we find that:

Variance Ratio= (σ\_max )^2/(σ\_min )^2 = 〖2.84〗2/〖1.20〗2 =5.6  
The correlation between the sample size and variance is -0.81.  
The coefficient of sample size variation (Δn) was calculated to be (SD(Group Sample Size))/(mean(Group Sample Size))=0.932  
According to the article’s conclusions with the parameters as calculated highlighted below (Table 10 of the article) the Type I error ratio will be liberal. However, most p-values reported by the two-way ANOVA are very small. The effects of unequal sample size and unequal variance within the sample groups should only affect relatively few term pairings. Assume p-values can be as much as four times larger than the given value. Variance ratio Pairing Coefficient of sample size variation Type I error rate 1.6, 1.7, 1.8 + or −1 0.50 Liberal  
2 −1 0.33; 0.50 Liberal  
3 1 0.50 Conservative  
−.50 0.50 Liberal  
−1 0.33; 0.50 Liberal  
**5** & 9 1 0.50 Conservative  
−.50 0.33; 0.50 Liberal  
**−1 0.16; 0.33; 0.50 Liberal**

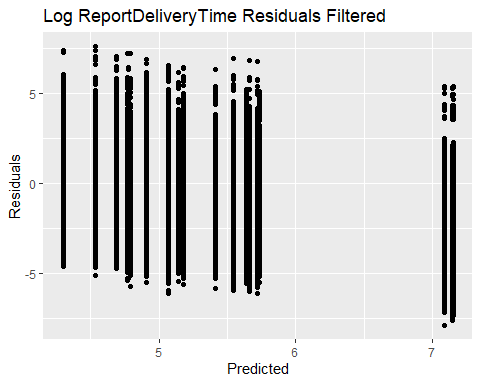
The following graph shows that the mean log of the delivery time varies by the time of day and has a different standard deviation during times of high loads.

ggplot(mapping=aes(x=HourBinned,y=Mean,color=Server,group=Server),data=sumstats.log.fltr)+  
 geom\_line()+  
 geom\_point()+  
 geom\_errorbar(aes(ymin=Mean-SD,ymax=Mean+SD),width=.1)+  
 ylab("Mean logged response of reports run") + xlab("Binned time report ran")+  
 ggtitle("Mean logged delivery time of reports ran by time of day per server")

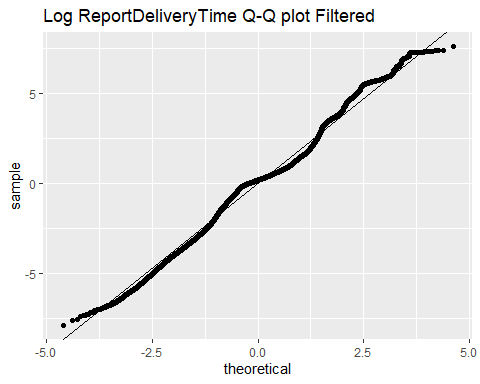


The following plots were created to further check for normality

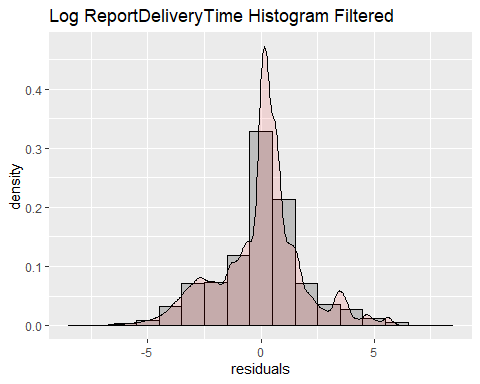
myfits<-data.frame(fitted.values=model.fit.HrBnSrvlg$fitted.values,residuals=model.fit.HrBnSrvlg$residuals)  
  
#Residual vs Fitted  
  
plot1.fltr.log<-ggplot(myfits,aes(x=fitted.values,y=residuals))+ylab("Residuals")+  
 xlab("Predicted")+geom\_point()+ggtitle("Log ReportDeliveryTime Residuals Filtered")  
  
#QQ plot of residuals #Note the diagonal abline is only good for qqplots of normal data.  
plot2.fltr.log<-ggplot(myfits,aes(sample=residuals))+  
 stat\_qq()+geom\_abline(intercept=mean(myfits$residuals), slope = sd(myfits$residuals))+  
 ggtitle("Log ReportDeliveryTime Q-Q plot Filtered")  
  
#Histogram of residuals  
plot3.fltr.log<-ggplot(myfits, aes(x=residuals)) +   
 geom\_histogram(aes(y=..density..),binwidth=1,color="black", fill="gray")+  
 geom\_density(alpha=.1, fill="red")+  
 ggtitle("Log ReportDeliveryTime Histogram Filtered")  
  
plot1.fltr.log



plot2.fltr.log



plot3.fltr.log



Residuals, Q-Q plot, and the Histogram all support the assumption of normality. Finally, a pairwise comparison based upon server and HourBinned was run. Remember to adjust p-value by a factor of four due to the unequal variance and sample sizes.

TukSrv <- TukeyHSD(model.fit.HrBnSrvlg,"Server",conf.level=.95)  
  
TukHrB <- TukeyHSD(model.fit.HrBnSrvlg,"HourBinned",conf.level=.95)  
  
TukSrvHrB <- TukeyHSD(model.fit.HrBnSrvlg,"Server:HourBinned",conf.level=.95)  
  
TukSrv[1]

## $Server  
## diff lwr upr p adj  
## SQLODR3-SQLODR2 -0.17335841 -0.19511648 -0.15160034 0.000000e+00  
## SQLODR6-SQLODR2 -0.07799528 -0.09852144 -0.05746913 2.142730e-14  
## SQLODR6-SQLODR3 0.09536313 0.07375766 0.11696859 2.853273e-14

TukHrB[1]

## $HourBinned  
## diff lwr upr p adj  
## 0809-0607 -1.48603606 -1.51545505 -1.45661707 0.0000000000  
## 1011-0607 -2.07262678 -2.11407317 -2.03118038 0.0000000000  
## 1213-0607 -2.62750071 -2.67392469 -2.58107674 0.0000000000  
## 1415-0607 -2.35477390 -2.39938694 -2.31016087 0.0000000000  
## 1617-0607 -1.97481181 -2.03203913 -1.91758450 0.0000000000  
## 1819-0607 -1.45780499 -1.49314890 -1.42246108 0.0000000000  
## 1011-0809 -0.58659071 -0.63150488 -0.54167655 0.0000000000  
## 1213-0809 -1.14146465 -1.19100922 -1.09192008 0.0000000000  
## 1415-0809 -0.86873784 -0.91658971 -0.82088598 0.0000000000  
## 1617-0809 -0.48877575 -0.54856240 -0.42898910 0.0000000000  
## 1819-0809 0.02823107 -0.01112208 0.06758423 0.3435666266  
## 1213-1011 -0.55487394 -0.61238038 -0.49736749 0.0000000000  
## 1415-1011 -0.28214713 -0.33820181 -0.22609244 0.0000000000  
## 1617-1011 0.09781496 0.03128098 0.16434895 0.0002939841  
## 1819-1011 0.61482179 0.56582182 0.66382176 0.0000000000  
## 1415-1213 0.27272681 0.21289781 0.33255581 0.0000000000  
## 1617-1213 0.65268890 0.58294542 0.72243238 0.0000000000  
## 1819-1213 1.16969572 1.11641928 1.22297217 0.0000000000  
## 1617-1415 0.37996209 0.31141073 0.44851346 0.0000000000  
## 1819-1415 0.89696892 0.84526286 0.94867497 0.0000000000  
## 1819-1617 0.51700682 0.45409292 0.57992073 0.0000000000

#TukSrvHrB[1]

Conclusion and discussion The business asked that we determine what are the best and worst times to run a report? And is there a difference in performance between servers. Due to the residual and Q-Q plot of the untransformed data it was found to be necessary to perform a log transform of the response data. The data responded well to the transformation displaying all the characteristics of normalized data. Using the table “Mean logged response of reports ran by time of day per server” above we can make the following recommendations. \* The number of reports run on server “SQLODR03” is about 20 % less than the other two servers. However, at most it has only 18.1% faster mean report delivery time(95% CI[5.3% , 32.4%]) this is between server “SQLODR03” and “SQLODR6” for the 10 AM through noon time period  
\* The time period 6 AM to 8 AM has the highest mean report delivery time, followed by 8 AM to 10 AM and 6 PM to 8 pm. Every effort should be made to not schedule new reports during these time frames.  
\* The time period noon to 2 PM has the lowest mean report delivery time, thus if possible new reports should be added there. The relative loading of the servers is as follows. Server “SQLODR02” has a 48.2% greater mean report delivery time (95% CI[30.7%, 68.1%]) than server”SQLODR06” for that time frame. While server “SQLODR3” has a 26.6% greater mean report delivery time (95% CI[11.2% , 44.2%]) than server ”SQLODR06”. Server ”SQLODR06” is the lightest loaded server as measured by mean report delivery time during the noon to 2PM time frame.  
\* The very late evening hours 8PM to midnight remain an option for scheduling reports. Care must be taken that any reports scheduled during that time do not run past midnight into the maintenance window.

ConvertPercent<-function(x){  
 result<-(exp(x)-1)\*100  
 return(result)  
}  
# these results are from running "TukSrvHrB[1]" They are used to create the first and third conclusions bullet points  
# SQLODR6:1011-SQLODR3:1011 0.166238080 0.051624515 0.2808516453 4.582670e-05  
# SQLODR6:1213-SQLODR2:1213 -0.393465747 -0.519129805 -0.2678016889 5.595524e-14  
# SQLODR6:1213-SQLODR3:1213 -0.236003508 -0.365704833 -0.1063021836 1.748803e-08  
  
z <- c(0.166238080, 0.051624515, 0.2808516453)  
  
z <- c(0.393465747, 0.519129805, 0.2678016889)  
z <- c(0.236003508, 0.365704833, 0.1063021836)  
  
ConvertPercent(z)

## [1] 26.61788 44.15297 11.21579