UNIVERSITY OF Waterloo

MSCI 719: Operations Analytics

Assignment 4: Vanderbilt University Medical Center Elective Surgery Prediction and Scheduling

Student: Advait Shah

Faculty: Engineering

Department: Management Sciences

Instructor: Prof. Hossein Abouee Mehrizi

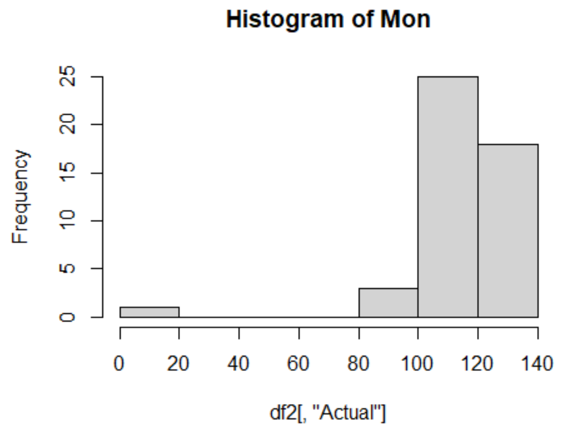
Contents

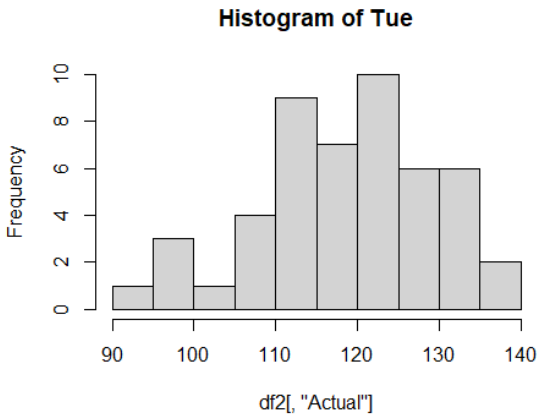
1. Comparison of weekdays

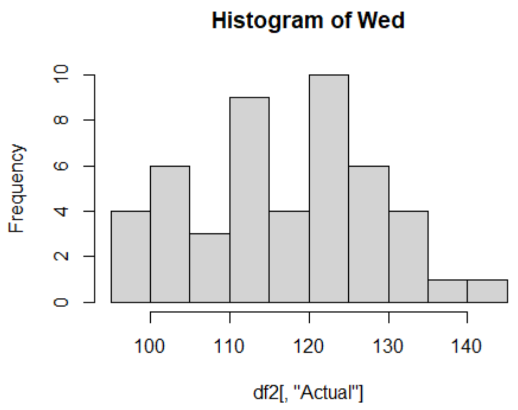
1.1. For each day of the week, plot the histogram of the actual number of surgeries

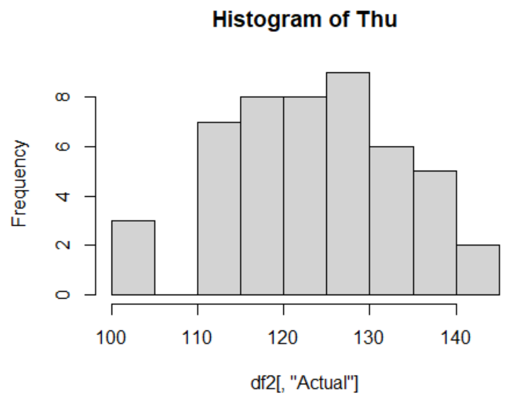
Histogram plots using R Programming:

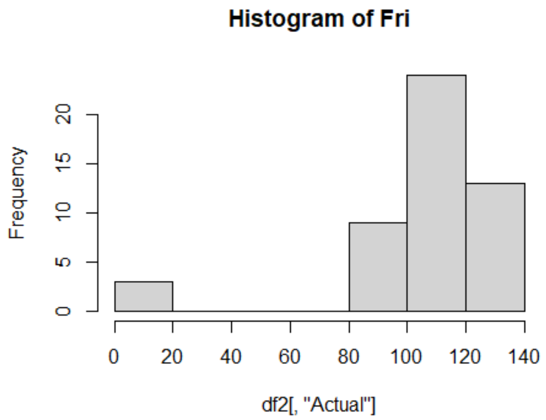
```
#a = read.csv(file.choose())

a = read.csv("Case5-data.csv")

head(a)

##     SurgDate DOW T...28 T...21 T...14 T...13 T...12 T...11 T...10 T...9 T...8
## 1 10-10-2011 Mon     38     45     60     63     65     70     73    73    73
## 2 11-10-2011 Tue     35     47     65     68     78     82     82    82    86
## 3 12-10-2011 Wed     26     43     54     62     72     72     72    74    87
## 4 13-10-2011 Thu     28     48     65     70     72     72     72    82    87
## 5 14-10-2011 Fri     31     40     50     50     50     54     62    68    71
## 6 17-10-2011 Mon     41     56     65     69     72     73     77    78    78
##   T...7 T...6 T...5 T...4 T...3 T...2 T...1 Actual
## 1    80    84    89    94    98   100   104    106
## 2    89    92    95    99    99    99   114    121
## 3    94    96   101   102   102   106   114    126
## 4    91    94    94    94    97    98   103    114
## 5    73    73    73    78    83    87    94    106
## 6    80    86    85    86    92    96   102    111

days=c("Mon", "Tue", "Wed", "Thu", "Fri")
for (i in 1:5){
    df2=a[a[,2]==days[i],]
    print(hist(df2[,"Actual"],main = paste("Histogram of" , days[i])))
}
```











1.2. Are the average and standard deviation for the number of surgeries on each day of the week (Monday to Friday) the same? Perform appropriate hypothesis tests and discuss the results:

We have performed single-factor Anova test, and checked sum of squares results for between groups and within groups to determine homogeneity of averages. And for testing homogeneity of variances, we applied Bartlett’s test.

Anova: Single Factor						
SUMMARY						
Groups	*Count*	*Sum*	*Average*	*Variance*	*Sd*	
Mon	47	5464	116.2553	340.629	18.45614	
Tue	49	5835	119.0816	118.0349	10.86439	
Wed	48	5618	117.0417	126.3387	11.24005	
Thu	48	5956	124.0833	107.7376	10.37967	
Fri	49	5175	105.6122	694.7007	26.35718	
ANOVA						
Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	8909.054	4	2227.264	8.002734	4.58E-06	2.409895
Within Groups	65681.83	236	278.3128			
Total	74590.88	240				

Variance and SD Homogeneity test: Bartlett's test:

First, we compute the k sample variances $s_1^2, s_2^2, \ldots, s_k^2$ from samples of size $n_1, n_2, \ldots, n_k$, with $\sum_{i=1}^{k} n_i = N$. Second, we combine the sample variances to give the pooled estimate

$$s_p^2 = \frac{1}{N-k} \sum_{i=1}^{k} (n_i - 1) s_i^2.$$

Now

$$b = \frac{[(s_1^2)^{n_1-1}(s_2^2)^{n_2-1} \cdots (s_k^2)^{n_k-1}]^{1/(N-k)}}{s_p^2}$$

is a value of a random variable B having the **Bartlett distribution**. For the special case where $n_1 = n_2 = \cdots = n_k = n$, we reject H_0 at the α-level of significance if

$$b < b_k(\alpha; n),$$

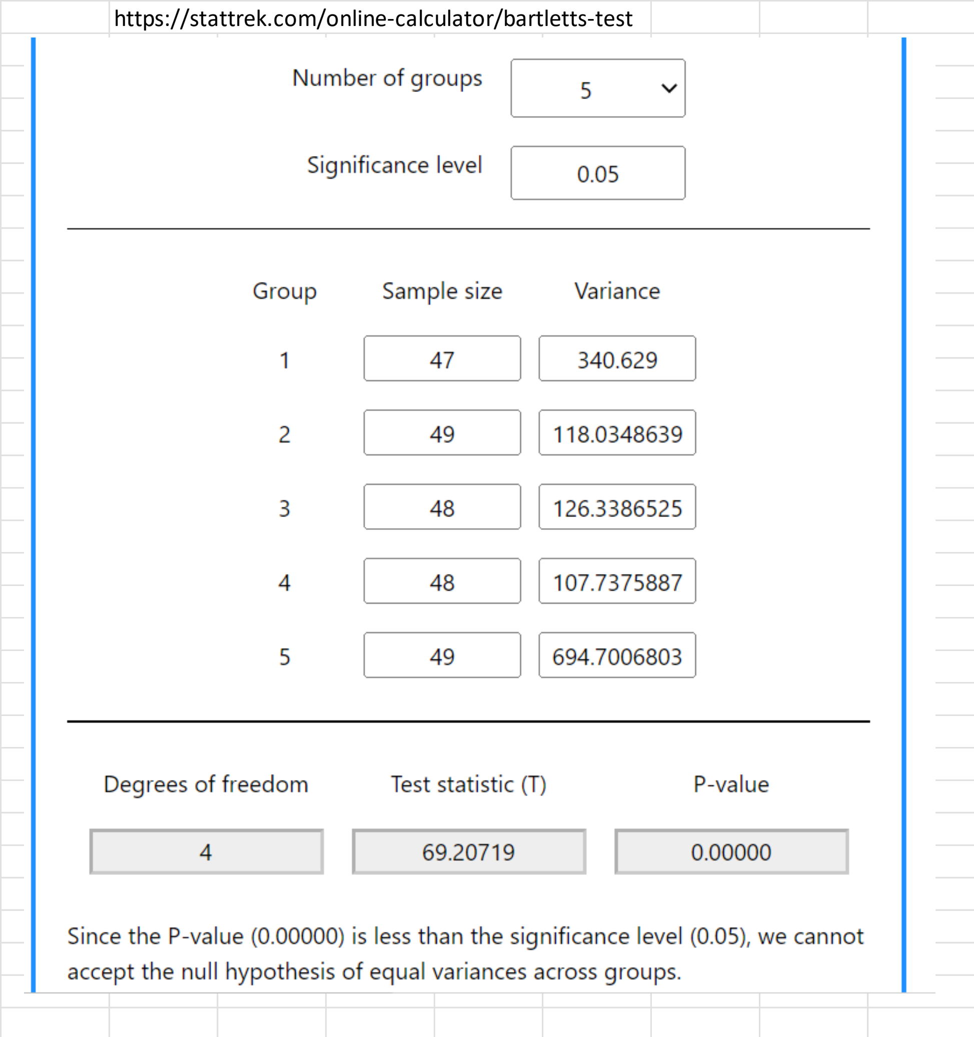
When the sample sizes are unequal, the null hypothesis is rejected at the α-level of significance if

$$b < b_k(\alpha; n_1, n_2, \ldots, n_k),$$

where

$$b_k(\alpha; n_1, n_2, \ldots, n_k) \approx \frac{n_1 b_k(\alpha; n_1) + n_2 b_k(\alpha; n_2) + \cdots + n_k b_k(\alpha; n_k)}{N}.$$

As before, all the $b_k(\alpha; n_i)$ for sample sizes $n_1, n_2, \ldots, n_k$ are obtained from Table A.10.



From Bartlett's test, we get p-value of less than 0.05. Therefore, H0 (Null hypothesis) has been rejected.

In both ANOVA and Bartlett's tests, we got P-values of less than 0.05, so our results are statistically significant. This suggests us to reject the null hypothesis of all groups having same or homogeneous averages and variances (and therefore also standard deviations). Also, the exact average and standard deviation values for each day of week are shown in the table.

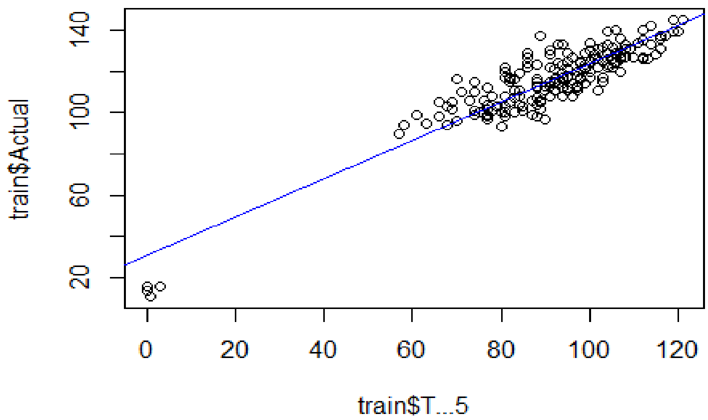
1.3. Is there a specific day of the week with a relatively higher average than the others? What could be the reason for the higher average?

It can be observed that Friday accounts for less average number of surgeries compared to other four working days of the week. Also, Thursday accounts for the highest average. Based on these two observations, we can comment that it is most likely that most emergency surgeries are being added on Thursday schedule, and surgeons tend to reduce the workload on Friday.

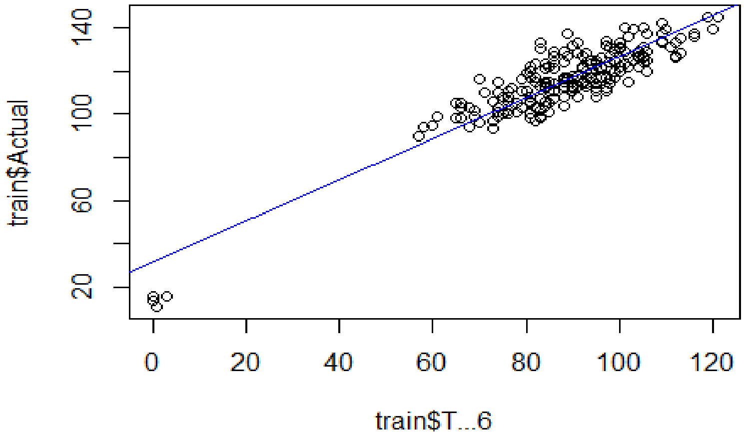
2. Longer prediction time and precision trade-off

Ajay Bose would like to predict the final number of surgeries on a specific day, using the data of scheduled surgeries. However, he doesn't know how many days before the desired date, he could predict the demand. Note that the sooner he predicts, the more error in prediction he will probably observe. Divide the data into 80 percent training and 20 percent testing. Consider the first model discussed in the class and calculate MSE (mean square error) on the test set for T – 5, T – 6, T – 7, T – 8, T – 9 as predictors. Visualize the data and discuss the trade-off between sooner prediction and an increase in error. Do the same steps for R2 values. Which day do you suggest as the predictor?

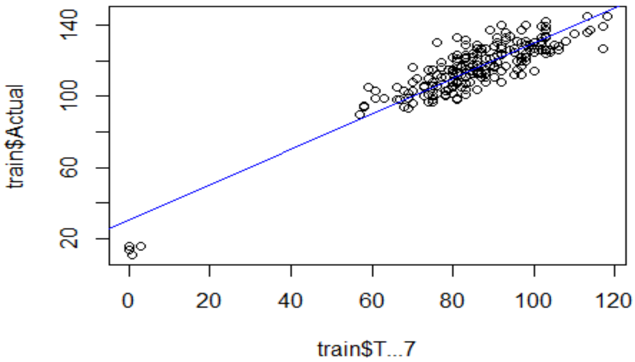
```
#make this example reproducible
set.seed(1)

#use 80% of dataset as training set and 20% as test set
sample <- sample(c(TRUE, FALSE), nrow(a), replace=TRUE, prob=c(0.8,0.2))
train  <- a[sample, ]
test   <- a[!sample, ]

t5_model = lm(Actual~T...5, train)
print(paste("T-5 based model R-Squared:",summary(t5_model)$r.squared))

## [1] "T-5 based model R-Squared: 0.827860438875018"

print(paste("T-5 based model MSE:",mean((test$Actual - predict.lm(t5_model
, test)) ^ 2)))

## [1] "T-5 based model MSE: 41.6998887704155"

plot(train$T...5,train$Actual)
abline(t5_model, col = "blue")
```




```
t6_model = lm(Actual~T...6, train)
print(paste("T-6 based model R-Squared:",summary(t6_model)$r.squared))

## [1] "T-6 based model R-Squared: 0.819831417075027"

print(paste("T-6 based model MSE:",mean((test$Actual - predict.lm(t6_model
, test)) ^ 2)))

## [1] "T-6 based model MSE: 40.7105698327936"

plot(train$T...6,train$Actual)
abline(t6_model, col = "blue")
```

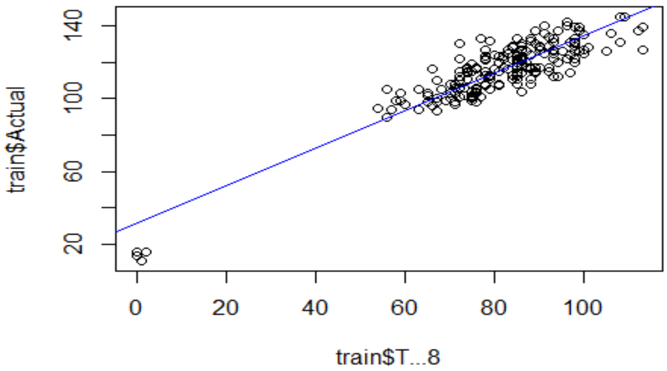



```
t7_model = lm(Actual~T...7, train)
print(paste("T-7 based model R-Squared:",summary(t7_model)$r.squared))

## [1] "T-7 based model R-Squared: 0.81812575168335"

print(paste("T-7 based model MSE:",mean((test$Actual - predict.lm(t7_model
, test)) ^ 2)))

## [1] "T-7 based model MSE: 47.9691965433956"

plot(train$T...7,train$Actual)
abline(t7_model, col = "blue")
```

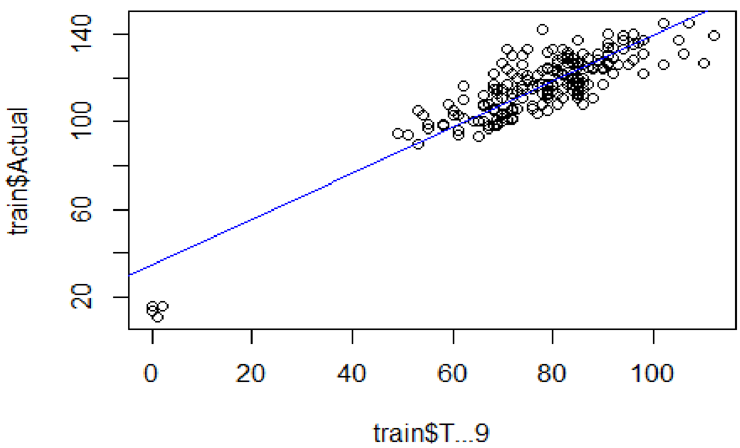



```
t8_model = lm(Actual~T...8, train)
print(paste("T-8 based model R-Squared:",summary(t8_model)$r.squared))

## [1] "T-8 based model R-Squared: 0.808648102575112"

print(paste("T-8 based model MSE:",mean((test$Actual - predict.lm(t8_model
, test)) ^ 2)))

## [1] "T-8 based model MSE: 58.2459247847008"
```

```
plot(train$T...8,train$Actual)
abline(t8_model, col = "blue")
```




```
t9_model = lm(Actual~T...9, train)
print(paste("T-9 based model R-Squared:",summary(t9_model)$r.squared))

## [1] "T-9 based model R-Squared: 0.785580610181517"

print(paste("T-9 based model MSE:",mean((test$Actual - predict.lm(t9_model
, test)) ^ 2)))

## [1] "T-9 based model MSE: 66.1200884449541"

plot(train$T...9,train$Actual)
abline(t9_model, col = "blue")
```



[1] “T-5 based model R-Squared: 0.827860438875018”

[1] “T-5 based model MSE: 41.6998887704155”

[1] “T-6 based model R-Squared: 0.819831417075027”

[1] “T-6 based model MSE: 40.7105698327936”

[1] "T-7 based model R-Squared: 0.81812575168335"

[1] "T-7 based model MSE: 47.9691965433956"

[1] "T-8 based model R-Squared: 0.808648102575112"

[1] "T-8 based model MSE: 58.2459247847008"

[1] "T-9 based model R-Squared: 0.785580610181517"

[1] "T-9 based model MSE: 66.1200884449541"

As we can see, T-5 and T-6 gives the least mean square error in predicting the actual surgeries when tested on the test dataset. whereas, T-7, T-8, T-9 predictors-based model gave comparatively higher prediction error. So, we could argue that it is a trade-off between accuracy level and prediction timing. We also plotted the best fit linear regression models based on each of these predictors, and calculated R-Squared values. It also suggests that linear regression model could be best fit on the T-5 and T-6 based predictors and thus, they gave better R-square values compared to T-7, T-8, T-9 predictors. This also explains why we have better prediction accuracy with T-5 and T-6 based models, as their linear regression models are more robust due to less bias in training dataset.

I would suggest T-6 based model to be used for predictions, because it has high prediction accuracy as well as team will have this information 6 days in advance of surgery, which would be sufficient in planning the schedule accurately about 6 days in advance of actual surgery days.

3. Time-Series vs. Regression

The provided data includes the number of surgeries scheduled to be performed on a specific date prior to the surgery (actual) date. As discussed in the lecture, there is a strong correlation between the predictor variables (columns in the data).

3.1. To reduce the correlation, consider add-on surgeries (the difference between two columns) as new predictors and develop a new regression model. Implement the following models and compare them with the models discussed in the lecture

- Model 1: Does not stratify by the day of the week.
- Model 2: Stratified by the day of the week.

```
# checking correlation between raw predictors
cor(a[,3:19])
```

```
##             T...28    T...21    T...14    T...13    T...12    T...11    T...10
## T...28 1.0000000 0.8947001 0.7669813 0.7612578 0.7642718 0.7696805 0.7442815
## T...21 0.8947001 1.0000000 0.8714275 0.8625057 0.8491198 0.8396694 0.8218751
## T...14 0.7669813 0.8714275 1.0000000 0.9755926 0.9403742 0.9188442 0.9134200
## T...13 0.7612578 0.8625057 0.9755926 1.0000000 0.9773372 0.9550263 0.9415538
## T...12 0.7642718 0.8491198 0.9403742 0.9773372 1.0000000 0.9866184 0.9620743
## T...11 0.7696805 0.8396694 0.9188442 0.9550263 0.9866184 1.0000000 0.9792885
## T...10 0.7442815 0.8218751 0.9134200 0.9415538 0.9620743 0.9792885 1.0000000
## T...9  0.7186066 0.8073506 0.9247739 0.9404125 0.9415326 0.9477643 0.9733221
## T...8  0.6978915 0.7946385 0.9199291 0.9311218 0.9221583 0.9181422 0.9351916
## T...7  0.6698647 0.7692789 0.9004517 0.9144495 0.9040636 0.8964697 0.9122042
## T...6  0.6694209 0.7713110 0.8901081 0.9119550 0.9128071 0.9064878 0.9185980
```

```
## T...5  0.6797108 0.7667651 0.8635365 0.8955542 0.9194134 0.9202565 0.9222467
## T...4  0.6854683 0.7662302 0.8460239 0.8782672 0.9109578 0.9239382 0.9279820
## T...3  0.6861281 0.7637451 0.8456964 0.8705655 0.8938988 0.9088628 0.9261966
## T...2  0.6550219 0.7429564 0.8481115 0.8627048 0.8769547 0.8856744 0.9079661
## T...1  0.6294322 0.7183636 0.8214784 0.8350405 0.8473868 0.8518777 0.8711997
## Actual 0.6082898 0.7024592 0.8008768 0.8127298 0.8187144 0.8198549 0.8421934
##            T...9     T...8     T...7     T...6     T...5     T...4     T...3
## T...28 0.7186066 0.6978915 0.6698647 0.6694209 0.6797108 0.6854683 0.6861281
## T...21 0.8073506 0.7946385 0.7692789 0.7713110 0.7667651 0.7662302 0.7637451
## T...14 0.9247739 0.9199291 0.9004517 0.8901081 0.8635365 0.8460239 0.8456964
## T...13 0.9404125 0.9311218 0.9144495 0.9119550 0.8955542 0.8782672 0.8705655
## T...12 0.9415326 0.9221583 0.9040636 0.9128071 0.9194134 0.9109578 0.8938988
## T...11 0.9477643 0.9181422 0.8964697 0.9064878 0.9202565 0.9239382 0.9088628
## T...10 0.9733221 0.9351916 0.9122042 0.9185980 0.9222467 0.9279820 0.9261966
## T...9  1.0000000 0.9715325 0.9550606 0.9456784 0.9333638 0.9258257 0.9245283
## T...8  0.9715325 1.0000000 0.9848287 0.9692359 0.9483349 0.9300646 0.9203496
## T...7  0.9550606 0.9848287 1.0000000 0.9845418 0.9600000 0.9383918 0.9255027
## T...6  0.9456784 0.9692359 0.9845418 1.0000000 0.9839807 0.9632278 0.9465638
## T...5  0.9333638 0.9483349 0.9600000 0.9839807 1.0000000 0.9849111 0.9643172
## T...4  0.9258257 0.9300646 0.9383918 0.9632278 0.9849111 1.0000000 0.9841580
## T...3  0.9245283 0.9203496 0.9255027 0.9465638 0.9643172 0.9841580 1.0000000
## T...2  0.9228736 0.9277084 0.9342842 0.9506493 0.9596923 0.9687847 0.9831174
## T...1  0.8951393 0.9092333 0.9181239 0.9279542 0.9373311 0.9431323 0.9509280
## Actual 0.8728896 0.8876746 0.8957787 0.8989004 0.9028267 0.9060397 0.9132423
##            T...2     T...1    Actual
## T...28 0.6550219 0.6294322 0.6082898
## T...21 0.7429564 0.7183636 0.7024592
## T...14 0.8481115 0.8214784 0.8008768
## T...13 0.8627048 0.8350405 0.8127298
## T...12 0.8769547 0.8473868 0.8187144
## T...11 0.8856744 0.8518777 0.8198549
## T...10 0.9079661 0.8711997 0.8421934
## T...9  0.9228736 0.8951393 0.8728896
## T...8  0.9277084 0.9092333 0.8876746
## T...7  0.9342842 0.9181239 0.8957787
## T...6  0.9506493 0.9279542 0.8989004
## T...5  0.9596923 0.9373311 0.9028267
## T...4  0.9687847 0.9431323 0.9060397
## T...3  0.9831174 0.9509280 0.9132423
## T...2  1.0000000 0.9700632 0.9364301
## T...1  0.9700632 1.0000000 0.9647269
## Actual 0.9364301 0.9647269 1.0000000
```

This suggests predictors are highly correlated.

```
#transforming columns to reduce correlation and create new predictors
df = data.frame(a$DOW,a$Actual)
for (i in 3:18){
  df[,i] <- a[,i+1] - a[,i]
}

head(df)
```

```
##   a.DOW a.Actual V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 V16 V17 V18
## 1   Mon      106  7 15  3  2  5  3  0   0   7   4   5   5   4   2   4   2
## 2   Tue      121 12 18  3 10  4  0  0   4   3   3   3   4   0   0  15   7
## 3   Wed      126 17 11  8 10  0  0  2  13   7   2   5   1   0   4   8  12
## 4   Thu      114 20 17  5  2  0  0 10   5   4   3   0   0   3   1   5  11
```

```
## 5   Fri      106  9 10  0  0  4  8  6   3   2   0   0   5   5   4   7  12
## 6   Mon      111 15  9  4  3  1  4  1   0   2   6  -1   1   6   4   6   9
```

```
cor(df[,2:18])
```

```
##             a.Actual            V3           V4           V5          V6
## a.Actual 1.00000000  0.4387188710  0.430460180  0.233514088  0.20064928
## V3       0.43871887  1.0000000000  0.085278214  0.075292587 -0.05653795
## V4       0.43046018  0.0852782138  1.000000000 -0.032837320 -0.12036956
## V5       0.23351409  0.0752925875 -0.032837320  1.000000000  0.29963241
## V6       0.20064928 -0.0565379465 -0.120369558  0.299632409  1.00000000
## V7       0.03972180 -0.1700832420 -0.151003310 -0.036153646  0.24055643
## V8       0.14382801  0.0540182054  0.113227291 -0.170295299 -0.25605335
## V9       0.12258045  0.0593385279  0.201505217 -0.237344504 -0.39855741
## V10      0.18992776  0.1101809539  0.123514207 -0.048779413 -0.17825376
## V11      0.20302726  0.0573891014  0.111827293  0.093094290 -0.03123994
## V12      0.26200727  0.1590979681  0.004177361  0.258653719  0.33282301
## V13      0.15485553 -0.0914336913 -0.186146694  0.268414589  0.61490297
## V14      0.04769927 -0.0533199467 -0.185790129 -0.005863455  0.22793634
## V15      0.07584365 -0.0207768508  0.036230015 -0.175276205 -0.24357969
## V16      0.11391471  0.0545345681  0.213964551 -0.253732554 -0.24487502
## V17      0.16774121 -0.0063629971 -0.025014011 -0.029635744 -0.04711121
## V18      0.10023360 -0.0008892634 -0.070056359 -0.054439086 -0.13748347
##                   V7          V8          V9          V10         V11
## a.Actual  0.03972180  0.14382801  0.12258045  0.189927759  0.20302726
## V3       -0.17008324  0.05401821  0.05933853  0.110180954  0.05738910
## V4       -0.15100331  0.11322729  0.20150522  0.123514207  0.11182729
## V5       -0.03615365 -0.17029530 -0.23734450 -0.048779413  0.09309429
## V6        0.24055643 -0.25605335 -0.39855741 -0.178253762 -0.03123994
## V7        1.00000000  0.11593586 -0.29620821 -0.257168308 -0.12782580
## V8        0.11593586  1.00000000  0.09596981 -0.158029789 -0.02084597
## V9       -0.29620821  0.09596981  1.00000000  0.193339581  0.18896773
## V10      -0.25716831 -0.15802979  0.19333958  1.000000000  0.06389516
## V11      -0.12782580 -0.02084597  0.18896773  0.063895156  1.00000000
## V12       0.04293635 -0.06668441 -0.33541318 -0.082325235  0.06007495
## V13       0.24692634 -0.26187287 -0.37100426 -0.177848747 -0.09941007
## V14       0.42852077  0.06082614 -0.33026216 -0.261193841 -0.12156296
## V15       0.06914415  0.36783426  0.01120868 -0.195192470 -0.10403560
## V16      -0.21333424  0.12602889  0.39087753  0.203840755  0.04889313
## V17      -0.10644765 -0.05780631  0.16516317  0.152398441  0.05226849
## V18      -0.08324249  0.04647092  0.10765362 -0.007550471 -0.03814337
##                   V12          V13          V14         V15         V16
## a.Actual  0.262007265  0.154855531  0.047699271  0.07584365  0.11391471
## V3        0.159097968 -0.091433691 -0.053319947 -0.02077685  0.05453457
## V4        0.004177361 -0.186146694 -0.185790129  0.03623001  0.21396455
## V5        0.258653719  0.268414589 -0.005863455 -0.17527620 -0.25373255
## V6        0.332823013  0.614902974  0.227936337 -0.24357969 -0.24487502
## V7        0.042936353  0.246926337  0.428520766  0.06914415 -0.21333424
## V8       -0.066684408 -0.261872870  0.060826136  0.36783426  0.12602889
## V9       -0.335413183 -0.371004256 -0.330262157  0.01120868  0.39087753
## V10      -0.082325235 -0.177848747 -0.261193841 -0.19519247  0.20384075
## V11       0.060074949 -0.099410070 -0.121562959 -0.10403560  0.04889313
## V12       1.000000000  0.282735968  0.006464538 -0.12370849 -0.13506213
## V13       0.282735968  1.000000000  0.172891048 -0.12757307 -0.26499132
```

```
## V14       0.006464538  0.172891048  1.000000000  0.15532127 -0.33978771
## V15      -0.123708493 -0.127573072  0.155321273  1.00000000 -0.05115462
## V16      -0.135062133 -0.264991319 -0.339787708 -0.05115462  1.00000000
## V17      -0.149643413  0.004412088 -0.076388266 -0.14887133  0.05637770
## V18      -0.173948762 -0.136714109 -0.062228387 -0.02059245  0.08136524
##                   V17           V18
## a.Actual  0.167741214  0.1002335955
## V3       -0.006362997 -0.0008892634
## V4       -0.025014011 -0.0700563594
## V5       -0.029635744 -0.0544390856
## V6       -0.047111210 -0.1374834693
## V7       -0.106447648 -0.0832424928
## V8       -0.057806310  0.0464709186
## V9        0.165163169  0.1076536235
## V10       0.152398441 -0.0075504711
## V11       0.052268494 -0.0381433746
## V12      -0.149643413 -0.1739487615
## V13       0.004412088 -0.1367141088
## V14      -0.076388266 -0.0622283875
## V15      -0.148871331 -0.0205924538
## V16       0.056377697  0.0813652380
## V17       1.000000000 -0.0380040786
## V18      -0.038004079  1.0000000000
```

The predictors are not correlated now, and hence we will include all of these predictors in our actual surgeries linear regression prediction.

```
#removing DOW from df for model 1

df1 = df[c(-1)]
head(df1)

##   a.Actual V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 V16 V17 V18
## 1      106  7 15  3  2  5  3  0   0   7   4   5   5   4   2   4   2
## 2      121 12 18  3 10  4  0  0   4   3   3   3   4   0   0  15   7
## 3      126 17 11  8 10  0  0  2  13   7   2   5   1   0   4   8  12
## 4      114 20 17  5  2  0  0 10   5   4   3   0   0   3   1   5  11
## 5      106  9 10  0  0  4  8  6   3   2   0   0   5   5   4   7  12
## 6      111 15  9  4  3  1  4  1   0   2   6  -1   1   6   4   6   9

model_1_not_str = lm(a.Actual~., df1)

summary(model_1_not_str)

## 
## Call:
## lm(formula = a.Actual ~ ., data = df1)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -27.0749  -6.7439   0.1262   5.7944  23.7427 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 28.26458    3.61133   7.827 1.97e-13 ***
```

```
## V3           1.26528    0.12082  10.472  < 2e-16 ***
## V4           1.09476    0.09467  11.563  < 2e-16 ***
## V5           1.17473    0.21123   5.561 7.60e-08 ***
## V6           1.29381    0.25981   4.980 1.27e-06 ***
## V7           1.22558    0.28702   4.270 2.89e-05 ***
## V8           0.76784    0.22280   3.446 0.000679 ***
## V9           1.09053    0.21554   5.059 8.75e-07 ***
## V10          1.19785    0.17721   6.760 1.18e-10 ***
## V11          0.84311    0.22037   3.826 0.000169 ***
## V12          1.28450    0.22854   5.620 5.63e-08 ***
## V13          1.12558    0.25647   4.389 1.76e-05 ***
## V14          0.94348    0.23563   4.004 8.47e-05 ***
## V15          1.31492    0.21856   6.016 7.21e-09 ***
## V16          0.59949    0.21436   2.797 0.005613 **
## V17          0.91463    0.14303   6.395 9.25e-10 ***
## V18          0.91519    0.12995   7.043 2.29e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.137 on 224 degrees of freedom
## Multiple R-squared:  0.7493, Adjusted R-squared:  0.7314
## F-statistic: 41.84 on 16 and 224 DF,  p-value: < 2.2e-16
```

So, we are getting 74.93 R-squared value in this linear regression fit.

```
# model 2: Stratified by the day of the week

DOW = c("Mon","Tue","Wed","Thu","Fri")

for (i in 1:5){
  df2 = df[df[,1]==DOW[i],]
  df2 = df2[c(-1)]
  model1 = lm(a.Actual~., df2)
  print(paste(DOW[i],"Multiple LR model R-squared value:",summary(model1)$
r.squared))

}

## [1] "Mon Multiple LR model R-squared value: 0.866348077861499"
## [1] "Tue Multiple LR model R-squared value: 0.646609148632659"
## [1] "Wed Multiple LR model R-squared value: 0.686860674014442"
## [1] "Thu Multiple LR model R-squared value: 0.572552488179656"
## [1] "Fri Multiple LR model R-squared value: 0.935184215065896"
```

Other Possible Models as discussed in the lecture:

Model 1 : Does not stratify by day of the week and use just T-7 as predictor

```
t7_model = lm(Actual~T...7, a)
print(paste("T-7 based LR model R-squared value:",summary(t7_model)$r.squa
red))

## [1] "T-7 based LR model R-squared value: 0.802419501289214"
```

Model 2 : Includes day of the week as dummy variables

```
t7d_model = lm(Actual~T...7+DOW, a)
print(paste("T-7+DOW based LR model R-squared value:",summary(t7d_model)$r
.squared))

## [1] "T-7+DOW based LR model R-squared value: 0.817761391970581"
```

Model 3 : stratify by day of the week and use just T-7 as predictor

```
DOW = c("Mon","Tue","Wed","Thu","Fri")

for (i in 1:5){
  df3 = a[a[,2]==DOW[i],]
  t7s_model = lm(Actual~T...7, df3)
  print(paste(DOW[i],"stratified T-7 based LR model R-squared value:",summ
ary(t7s_model)$r.squared))

}

## [1] "Mon stratified T-7 based LR model R-squared value: 0.824985517101034"
## [1] "Tue stratified T-7 based LR model R-squared value: 0.537466514873414"
## [1] "Wed stratified T-7 based LR model R-squared value: 0.668428942481873"
## [1] "Thu stratified T-7 based LR model R-squared value: 0.615438510844795"
## [1] "Fri stratified T-7 based LR model R-squared value: 0.915886722095723"
```

Thus, by comparing the models formed after data transformation vs. models based on T-7, we could say that new models give comparatively better fit and R-squared value. However, for this model to work, its predictors require data of daily addition in scheduled surgeries even close to surgery day. So, these models would not be much helpful in predicting surgeries in advance.

3.2. Consider the surgery (actual) date as a time series with September 4th to September 14th as the testing set and the rest as the training set. Fit a Moving Average (MA) model to the time series and visualize it

In the Moving Average (MA) approach, to forecast number of surgeries on the next day, we use following formula,

$$F_{t+1} = \frac{1}{N} \sum_{k=t+1-N}^{t} Y_k$$

Where, F_{t+1}: Forecasted surgeries at time t+1 and Y_k: Actual Surgeries at time k

So, based on this approach, we could predict number of surgeries on the ongoing basis. And then to evaluate our prediction, we will use following error measures:

$$\frac{\sum_{t=1}^{n}(Y_t - F_t)^2}{n}$$

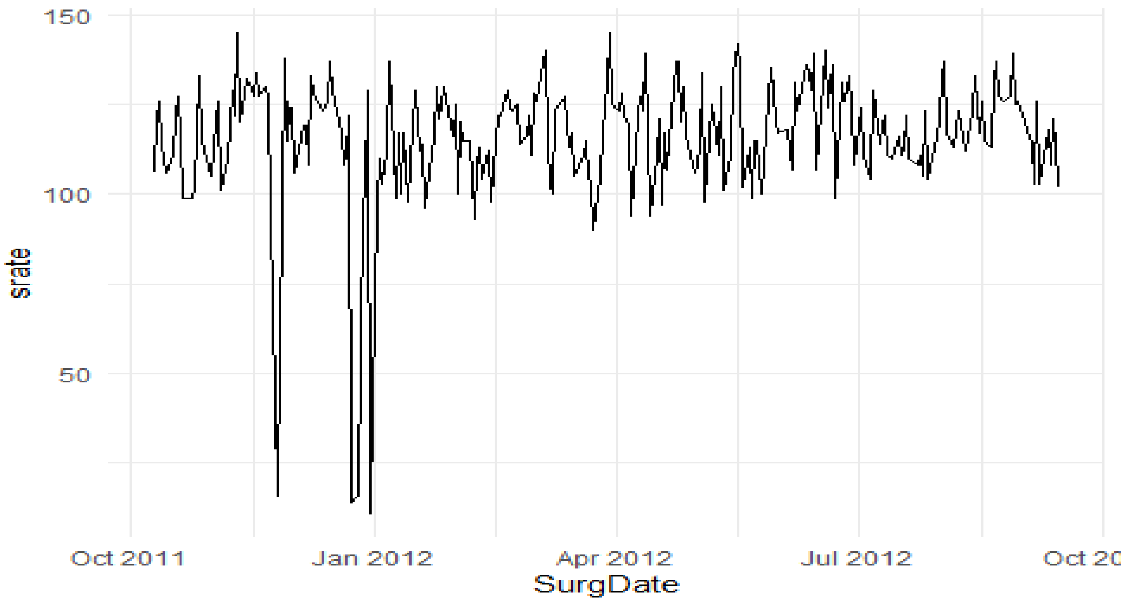
MSE: the Mean Squared Error between forecast and actual:

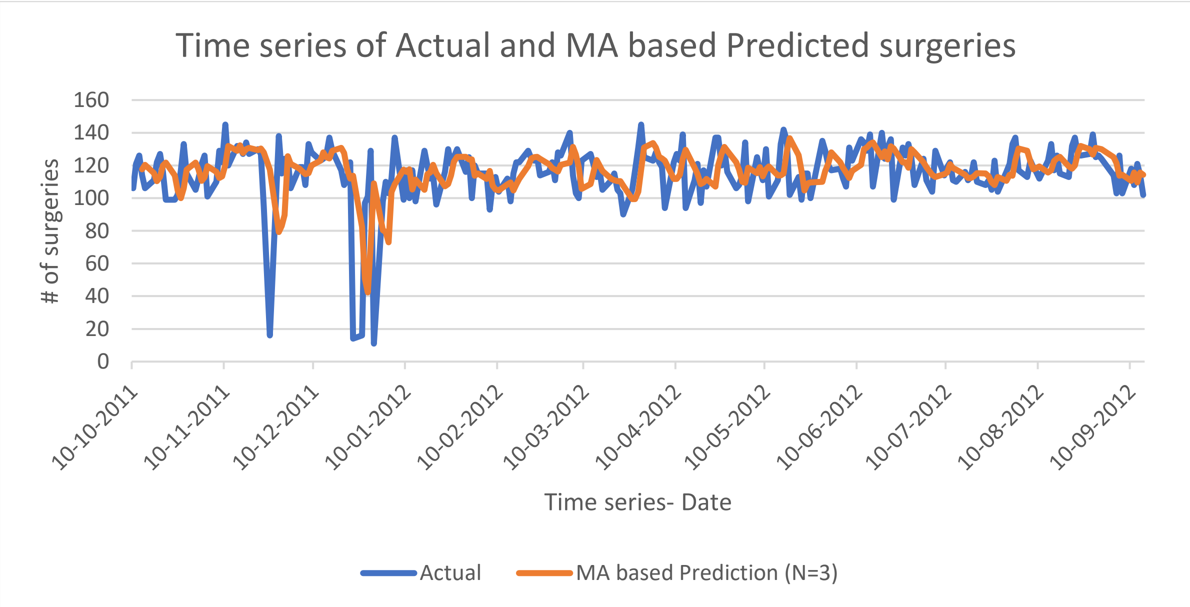
Then, we can train model based on different values of N (i.e. number of prior periods used for moving average calculation and forecast) and measure these errors and select best N, hyper parameter, which minimizes our MSE error loss function.

After trying with N=2 to N=10, we observe that with N=3, we get the following minimum forecasting error results for our prediction test period:

MSE = 125.9506

```
library(dplyr)

library(magrittr)
a %<>%
  mutate(SurgDate= as.Date(SurgDate, format= "%d-%m-%Y"))

train <- subset(a, SurgDate < "2012-09-04")
test <- subset(a, SurgDate >= "2012-09-04")

tail(train)

##       SurgDate DOW T...28 T...21 T...14 T...13 T...12 T...11 T...10 T...9 T...8
## 227 2012-08-24 Fri     29     34     67     67     67     67     75    91    95
## 228 2012-08-27 Mon     40     44     66     69     79     82     85    85    86
## 229 2012-08-28 Tue     34     56     69     84     91     94     94    94    99
## 230 2012-08-29 Wed     36     57     76     81     87     87     87    92    99
## 231 2012-08-30 Thu     29     59     86     88     88     88     97   102   105
## 232 2012-08-31 Fri     19     38     58     58     58     62     68    71    80
##     T...7 T...6 T...5 T...4 T...3 T...2 T...1 Actual
## 227   104   104   104   108   115   119   126    126
## 228    92    98   107   109   111   116   123    127
## 229   103   110   119   124   125   128   139    139
## 230   101   102   104   103   103   107   114    125
## 231   106   112   113   113   113   115   124    126
## 232    86    86    86    94    93    99   116    124

library(zoo)

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

s <- a %>%
  select(SurgDate, srate = Actual) %>%
  mutate(srate_tma = rollmean(srate, k = 3, fill = NA, align = "right"))

library(ggplot2)

## Warning: package 'ggplot2' was built under R version 4.1.3

ggplot(s, aes(SurgDate,srate)) +
  geom_line() +
  theme_minimal()
```





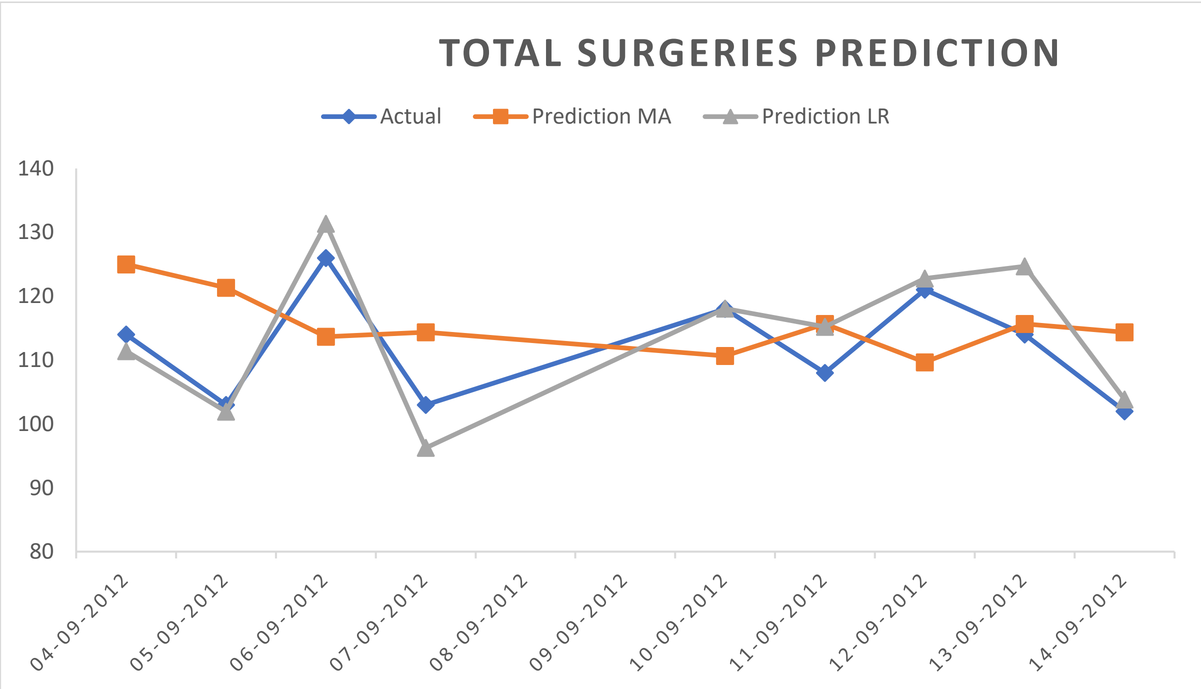
3.3. Compare the result of the regression model with the MA model visually and based on MSE. Which model provides a better prediction? What could be the potential reason?

Now, to compare MSE of Moving average model prediction with Linear regression model, let's consider linear regression model with the least MSE on test data. As we saw in section 2, T-6 based model gave us least prediction error, so let's use this model and see MSE for the required time period prediction.

```
# let's use T-6 based model, which had given us least MSE of prediction on
test data
summary(t6_model)

##
## Call:
```

```
## lm(formula = Actual ~ T...6, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -21.711  -5.370  -0.689   5.068  22.534
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 31.76291    2.84760   11.15   <2e-16 ***
## T...6        0.94823    0.03167   29.94   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.003 on 197 degrees of freedom
## Multiple R-squared:  0.8198, Adjusted R-squared:  0.8189
## F-statistic: 896.4 on 1 and 197 DF,  p-value: < 2.2e-16

print(paste("T-6 based model MSE:",mean((test$Actual - predict.lm(t6_model
, test)) ^ 2)))

## [1] "T-6 based model MSE: 28.2909188554732"
```

So, with linear regression, we get just **28.29** MSE error, whereas, we had **125.95** MSE error in the Moving average model-based predictions. We can also visualize our predictions based on these two models as below.



From this graph and MSE results, it is clear that Linear regression (LR) model is much better than the Moving average (MA) model. The main reason for this can be stated as the amount of information both models use in the prediction task. Linear regression model uses information of scheduled surgeries 6 days before the actual surgery day and scale it appropriately with past data-based trained linear model's weights for accurate predictions. Whereas, Moving average model is just considering average of past 3 days as prediction, and this does not work well, because there is huge variance in average surgeries on each day of week, and due to this reason prediction is not accurate as it is does not account for day of week into prediction, as well as it does not consider data regarding actual planned or scheduled surgeries for that day. So, this results into poor predictions from the MA model; whereas, LR model seems to give fairly good prediction on the daily number of surgeries.

References:

[1] H A Mehrizi, eBook: MSCI 719 Winter 2023 Cases Multiple (ID: 9723713) Accessed: Jan. 22, 2023. [Online].

Available:
https://www.campusebookstore.com/integration/AccessCodes/default.aspx?permalinkId=ee044bf2-fe82-4db0-ad22-088e81954eef&frame=YES&t=permalink&sid=4u2faw45zyslbp45bbqlpc55