How Can the Hospital Manage Admission Delays for Emergency Patients?

Analysis of Admission Delays for Emergency Ward Patients

John Karuitha

Friday April 21, 2023

Contents


```
knitr::opts_chunk$set(echo = TRUE, message = FALSE, warning = FALSE)

if (!require(pacman)) {
  install.packages("pacman")
}
```

```
## Loading required package: pacman
```

```
pacman::p_load(
  tidyverse, janitor, skimr,
  kableExtra, readxl, conflicted,
  GGally, corrplot, ggthemes,
```

```
  randomForest, vip, stargazer,
  naivebayes, e1071, caret,
  kernlab
)

options(digits = 3)
theme_set(ggthemes::theme_clean())
```

1 Summary

In this analysis, I have used emergency department data from a fictitious hospital to generate insights to reduce waiting times. Following my analysis, I come up with the following insights.

- The critical factor in wait times are the number of patients.
- The number of patients vary by number of day or day of week.

My recommendations are as follows;

- Vary the number of staff depending on days of week that have more emergency cases.
- Vary staff by hour of day, with more staff during hours with more emergency cases like 1100 hrs, and less during times when there are fewer cases like around midnight.
- Given that the hospital has fixed number of beds, there should be arrangements to create portable, temporary emergency wards and beds to cater for the upsurge in emergency cases duting particular days of week and hours of day.

2 Background

In hospital emergency wards, the longer the patient waits in a queue before being attended, the greater the risk of mortality. In this analysis, I use data regarding the patient wait times in an emergency department in an anonymous hospital to derive insights that can help the hospital management reduce the wait times. Specifically, the analysis aims uncovering factors associated with admission delays so as to advise the management of actionable strategies to reduce the delays.

To achieve this goal, I run the following models for the analysis.

- Simple Linear Regression Model.
- Support Vector Machines (SVM) Model.
- Naive Bayes Model.
- Random Forest Model.

But first, I load and explore the data, run the models, and, finally, make recommendations. The analysis then concludes.

3 Data

3.1 *Overview of the Data*

In this section, I load and explore the data.

```
## Load the data ----
waiting <- readxl::read_xlsx("Hw_data.xlsx") |>
  janitor::clean_names()

## Overview of the data
glimpse(waiting)
```

```
## Rows: 744
## Columns: 8
## $ date_and_time                         <dttm> 2016-05-01 00:00:00, 2016-05-01 ~
## $ number_of_ed_beds                     <dbl> 72, 72, 72, 72, 72, 72, 72, 72, 7~
## $ number_of_ip_beds                     <dbl> 532, 532, 532, 532, 532, 532, 532~
## $ number_of_ed_pts                      <dbl> 57, 57, 52, 40, 40, 34, 26, 28, 2~
## $ number_of_ed_pts_waiting_ip_bed       <dbl> 9, 8, 9, 9, 9, 9, 11, 10, 12, 11,~
## $ number_of_critical_care_pts_display  <dbl> 3.5, 5.5, 5.5, 5.5, 6.5, 5.5, 6.0~
## $ door_to_bed_time_for_last_ed_patient <dbl> 1.82, 1.05, 1.10, 1.56, 0.01, 0.1~
## $ longest_admit_time_waiting_in_ed     <dbl> 3.55, 4.25, 6.91, 1.49, 2.11, 3.3~
```

The data consists of 744 observations of 8 variables. Table 1 below describes the variables.

```
# Describing the variables in the raw data ----
tribble(
  ~Variable, ~Description,
  "date_and_time",
  "Time and date of patient arrival.",
  "number_of_ed_beds",
  "Number of beds in the emergency department.",
  "number_of_ip_beds",
  "Number of inpatient beds.",
  "number_of_ed_pts",
  "Number of emergency department patients.",
  "number_of_ed_pts_waiting_ip_bed",
```

```
  "Number of emergency department patients waiting for bed",
  "number_of_critical_care_pts_display",
  "Number of critical care patients display",
  "door_to_bed_time_for_last_ed_patient",
  "Door to bed time for last patient in emergency department.",
  "longest_admit_time_waiting_in_ed",
  "Longest admission time for patient waiting in emergency department."
) |>
  kbl(booktabs = TRUE, caption = "Variables Description") |>
  kable_classic(
    full_width = TRUE,
    latex_options = "hold_position"
  )
```

Table 1: Variables Description

Variable	Description
date_and_time	Time and date of patient arrival.
number_of_ed_beds	Number of beds in the emergency department.
number_of_ip_beds	Number of inpatient beds.
number_of_ed_pts	Number of emergency department patients.
number_of_ed_pts_waiting_ip_bed	Number of emergency department patients waiting for bed
number_of_critical_care_pts_display	Number of critical care patients display
door_to_bed_time_for_last_ed_patient	Door to bed time for last patient in emergency department.
longest_admit_time_waiting_in_ed	Longest admission time for patient waiting in emergency department.

The target variable (dependent variable) is the longest admit time waiting in emergency department (longest_admit_time_waiting_in_ed).

3.2 *Missing and Duplicate values*

The data has no missing values.

```
## Check missing values
waiting |>
  sapply(is.na) |>
  colSums() |>
  tibble(variables = names(waiting), missing = _) |>
  kbl(booktabs = TRUE, caption = "Missing values") |>
  kable_classic(
    full_width = FALSE,
    latex_options = "hold_position"
  )
```

Likewise, the data has no duplicate rows.

```
## Check duplicate observations
waiting |>
  janitor::get_dupes()
```

```
## # A tibble: 0 x 9
## # i 9 variables: date_and_time <dttm>, number_of_ed_beds <dbl>,
```

Table 2: Missing values

variables	missing
date_and_time	0
number_of_ed_beds	0
number_of_ip_beds	0
number_of_ed_pts	0
number_of_ed_pts_waiting_ip_bed	0
number_of_critical_care_pts_display	0
door_to_bed_time_for_last_ed_patient	0
longest_admit_time_waiting_in_ed	0

```
## #   number_of_ip_beds <dbl>, number_of_ed_pts <dbl>,
## #   number_of_ed_pts_waiting_ip_bed <dbl>,
## #   number_of_critical_care_pts_display <dbl>,
## #   door_to_bed_time_for_last_ed_patient <dbl>,
## #   longest_admit_time_waiting_in_ed <dbl>, dupe_count <int>
```

Given that this is a relatively clean dataset, I embark on data exploration.

3.3 *Exploratory Data Analysis*

I start by summarising the data.

```
## Summary statistics for the data
waiting |>
  select(-date_and_time) |> skimr::skim_without_charts() |>
  select(-complete_rate, -n_missing) |>
  set_names(c(
    "no", "variable", "Mean", "SD", "Min",
    "Q1", "Median", "Q3", "Max"
  )) |>
  select(-no) |> kbl(booktabs = TRUE, caption = "Data Summary") |>
  kable_classic(
    full_width = FALSE, latex_options = "hold_position"
  )
```

Table 3: Data Summary

variable	Mean	SD	Min	Q1	Median	Q3	Max
number_of_ed_beds	72.000	0.00	72.00	72.00	72.00	72.000	72.00
number_of_ip_beds	532.000	0.00	532.00	532.00	532.00	532.000	532.00
number_of_ed_pts	60.579	19.57	20.00	43.00	62.00	76.000	109.00
number_of_ed_pts_waiting_ip_bed	10.633	5.24	0.00	6.00	10.00	14.000	29.00
number_of_critical_care_pts_display	5.443	2.35	0.00	3.50	5.50	7.000	14.50
door_to_bed_time_for_last_ed_patient	0.761	1.16	-0.07	0.03	0.23	0.883	6.76
longest_admit_time_waiting_in_ed	5.041	5.24	0.00	1.63	3.36	6.562	30.02

I do a pairs plot for the variables to uncover the significant relationships between the variables.

```r
## Pairs plotof the raw data
waiting |>
  select(
    -date_and_time, -number_of_ed_beds,
    -number_of_ip_beds
  ) |>
  GGally::ggpairs(title = "Variables Pairs Plots")
```

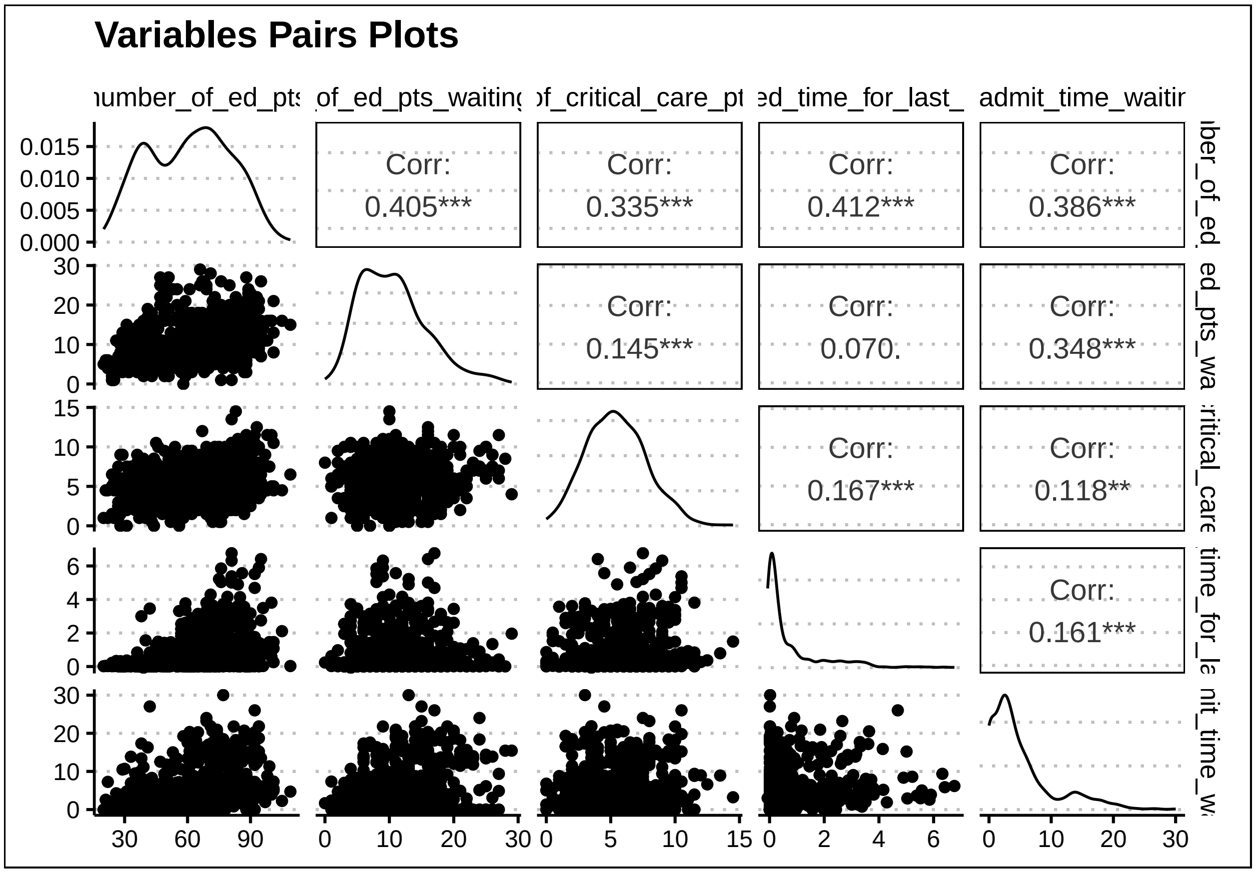

Figure 1: Pairs Plot for Raw Data

Note that the number of both inpatient and emergency department beds are constant and hence the NA value for the correlations of these variables and other variables in the dataset. Hence, I omit both variables in the correlation analysis.

Overall, there are significant correlations between waiting times and the number of patients (number_of_ed_pts), and the patients waiting in inpatient beds (number_of_ed_pts_waiting_ip_bed). The variables can be useful in explaining waiting times in emergency rooms except for the number of beds.

I remove the two constant columns from the data; number_of_ed_beds, number_of_ip_beds.

```
## Remove the columns with constant values
## These are; number_of_ed_beds, number_of_ip_beds.
waiting <- waiting |>
  remove_constant()
```

3.4 *Feature Engineering*

Given that the flow of patients may vary by day of the week and time of day, I split the date_and_time variable into two variables, weekday (Sunday (1) to Saturday (7)) and hour of the day (in 24 hour format 00 to 23). Note that the data was collected in May 2016 hence the month and year may not be useful in the analysis.

Given that I also run Naive Bayes and Support Vector Machines (SVM) models, I create a new categorical variable from the wait times. Specifically, I segment the target variable (longest_admit_time_waiting_in_ed) into 9 categories. Examples of these categories are patients that wait for 0-3.34 minutes, 3.34 to 6.67 minutes, and so on. I have used my own judgement to select these segments.

```
## Creating new variables, day of week, hour of day, and wait_cat
## wait_cat bins the waiting times into 9 categories
waiting <- waiting |>
  mutate(
    week_day = wday(date_and_time, label = FALSE),
    hour_day = hour(date_and_time)
  ) |>
  select(-date_and_time) |>
  mutate(
    hour_day = factor(hour_day),
    week_day = factor(week_day)
  ) |>
  mutate(wait_cat = cut(longest_admit_time_waiting_in_ed,
    breaks = 9
  ))
```

4 Analysis

4.1 Distribution of Wait Times

Figure 1 below shows the distribution of weight times. We see that the variable is skewed with much of the wait time less than 10 minutes. However, there are outliers of upto 30 minutes which could mean life and death for patients. Next, we see the distribution of patients based on the waits times for the categorical variable `wait_cat` (Figure 2).

```r
## Distribution of patient wait times
waiting |> ggplot(aes(x = longest_admit_time_waiting_in_ed)) +
  geom_histogram(color = "black", fill = "gray80") +
  geom_density(aes(y = ..count..)) +
  labs(x = "Wait Time", y = "Count",
    title = "Distribution of Wait Time")
```

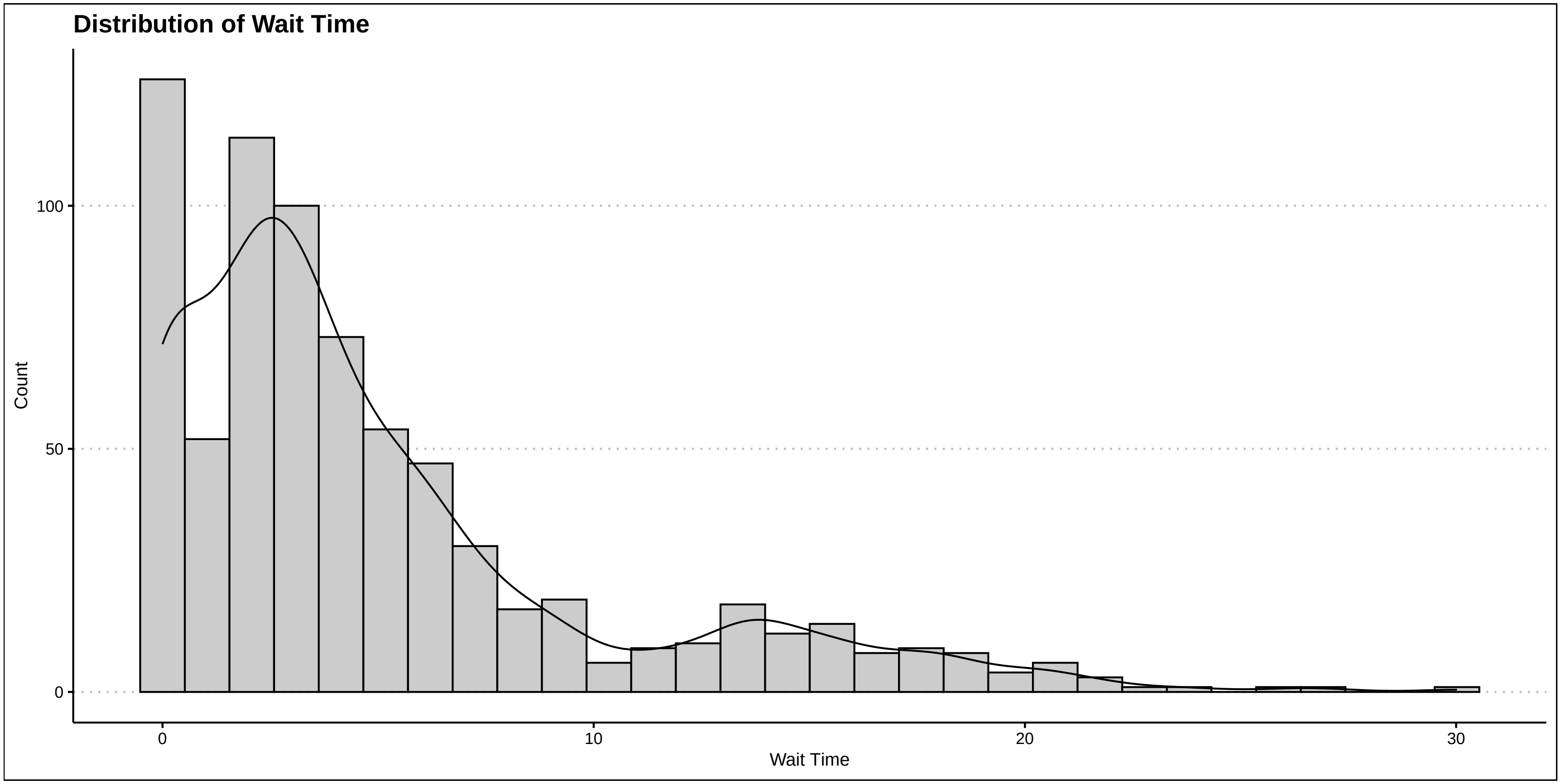

Figure 2: Distribution of Patient Wait Times

```r
## Waiting time by categories
waiting |> count(wait_cat) |>
  ggplot(mapping = aes(x = wait_cat, y = n)) + geom_col() +
  labs(x = "length of Wait", y = "Count",
  title = "Waiting Times in Emergency Departments") + coord_flip()
```

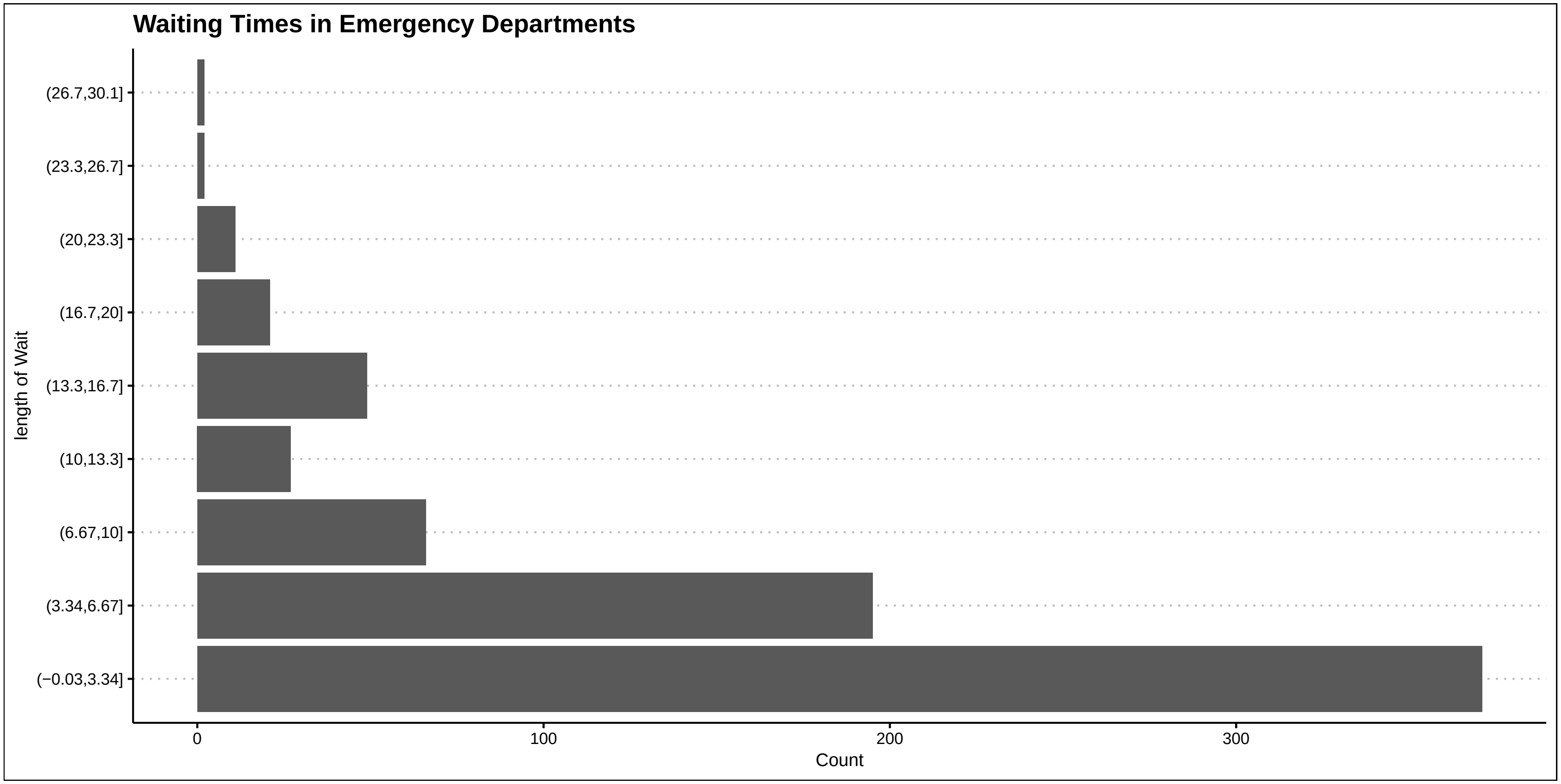

Figure 3: Waiting Times in Emergency Departments

Again the skewness in the histogram repeats, with most waits between zeo and 4 minutes. However, there are outliers of upto thirty minutes which are a cause for concern.

4.2 Day of Week and Hour of Day with Most emergencies

Table 4 below shows the distribution of cases by week day. The table shows that most cases occur on Sundays, Mondays, and Tuesdays with 120 admissions each.

```
## Patients by day of the week
waiting |>
  count(week_day) |>
  kbl(booktabs = TRUE, caption = "Patients by Day of Week") |>
  kable_classic(
    full_width = FALSE,
    latex_options = "hold_position"
  )
```

Table 4: Patients by Day of Week

week_day	n
1	120
2	120
3	120
4	96
5	96
6	96
7	96

For hour of day, there are a uniform number of admissions for this data (Table 5).

```
## Patients by hour of day
waiting |>
  count(hour_day) |>
  kbl(booktabs = TRUE, caption = "Patients by Hour of Day") |>
  kable_classic(
    full_width = FALSE,
    latex_options = "hold_position"
  )
```

4.3 Models

In this section, I run the models in the following order.

1. Linear Regression Model.
2. Support Vector Machines Model.
3. Naive Bayes Model.
4. Random forest Model.

4.3.1 *Linear Regression Model.*

The regression analysis [@]Muller2016 shows the factors that are important in determining waiting times for patients.

Table 5: Patients by Hour of Day

hour_day	n
0	31
1	31
2	31
3	31
4	31
5	31
6	31
7	31
8	31
9	31
10	31
11	31
12	31
13	31
14	31
15	31
16	31
17	31
18	31
19	31
20	31
21	31
22	31
23	31

```r
## Run the regression model
lm_model <- lm(longest_admit_time_waiting_in_ed ~ . - wait_cat, data = waiting)
```

```r
## Generate regression model output for either HTML or PDF output

if (knitr::is_html_output()) {

  stargazer::stargazer(lm_model,
  type = "latex",
  title = "Regression Output"
)
} else {

  stargazer::stargazer(lm_model,
  type = "latex",
  title = "Regression Output",
  font.size = "tiny"
)
}
```

% Table created by stargazer v.5.2.3 by Marek Hlavac, Social Policy Institute. E-mail: marek.hlavac at gmail.com % Date and time: Fri, Apr 21, 2023 - 08:11:33 PM

Table 6: Regression Output

	Dependent variable:
	longest_admit_time_waiting_in_ed
number_of_ed_pts	0.026
	(0.020)
number_of_ed_pts_waiting_ip_bed	0.109**
	(0.043)
number_of_critical_care_pts_display	−0.030
	(0.075)
door_to_bed_time_for_last_ed_patient	0.207
	(0.156)
week_day2	−0.451
	(0.588)
week_day3	0.896
	(0.632)
week_day4	0.493
	(0.619)
week_day5	2.140***
	(0.641)
week_day6	0.812
	(0.626)
week_day7	−0.448
	(0.606)
hour_day1	−0.377
	(1.120)
hour_day2	−0.010
	(1.140)
hour_day3	−0.421
	(1.170)
hour_day4	−0.593
	(1.200)
hour_day5	−0.037
	(1.220)
hour_day6	−0.249
	(1.230)
hour_day7	−0.503
	(1.230)
hour_day8	0.647
	(1.190)
hour_day9	−0.642
	(1.140)
hour_day10	1.550
	(1.130)
hour_day11	3.750***
	(1.120)
hour_day12	4.740***
	(1.130)
hour_day13	5.110***
	(1.130)
hour_day14	6.430***
	(1.140)
hour_day15	4.820***
	(1.140)
hour_day16	4.990***
	(1.150)
hour_day17	3.640***
	(1.150)
hour_day18	2.900**
	(1.150)
hour_day19	0.473
	(1.170)
hour_day20	1.580
	(1.170)
hour_day21	−0.220
	(1.170)
hour_day22	−0.201
	(1.140)
hour_day23	−0.116
	(1.120)
Constant	0.326
	(1.300)
Observations	744
R^2	0.339
Adjusted R^2	0.308
Residual Std. Error	4.360 (df = 710)
F Statistic	11.000*** (df = 33; 710)
Note:	*p<0.1; **p<0.05; ***p<0.01

The critical factors include;

- Number_of_ed_patients_waiting_in_ip_beds.
- Weekday.
- Hour of day.

Specifically, the number of patients waiting in inpatient beds has a direct relationship with waiting times. Besides, some days of the week are likely to experience longer waiting times than others. For instance, compared to Sunday (day 1), Thursday(day 5) has longer waiting times. Likewise, some hours of the day see longer waiting times. As case in point, 1100 hrs has a higher waiting times compared to midnight.

Recommendation: Increase the number of staff during peak days like Thursday so as to clear the number of patients waiting in inpatient beds.

4.3.2 *Support Vector Machines Model.*

```
## Create a dataset for Naive Bayes and SVM Models

final_data <- waiting |>
  select(-longest_admit_time_waiting_in_ed) |>
  data.frame()
```

We run the SVM model and generate variable importance plots (DataCamp n.d.). As in the Naive Bayes case, for patients that wait between 20 and 23.3 minutes, the number of patients is critical, followed by the door to bed time for the last emergency department patient.

```r
# Create cross validation folds
train_control <- trainControl(
  method = "repeatedcv", number = 10, repeats = 3)
# Fit the model
svm_model <- train(wait_cat ~ ., data = final_data,
  method = "svmLinear", trControl = train_control,
  preProcess = c("center", "scale"), tuneGrid = expand.grid(C = seq(0, 2,
    length = 20
  )))
# View the model
plot(varImp(svm_model))
```

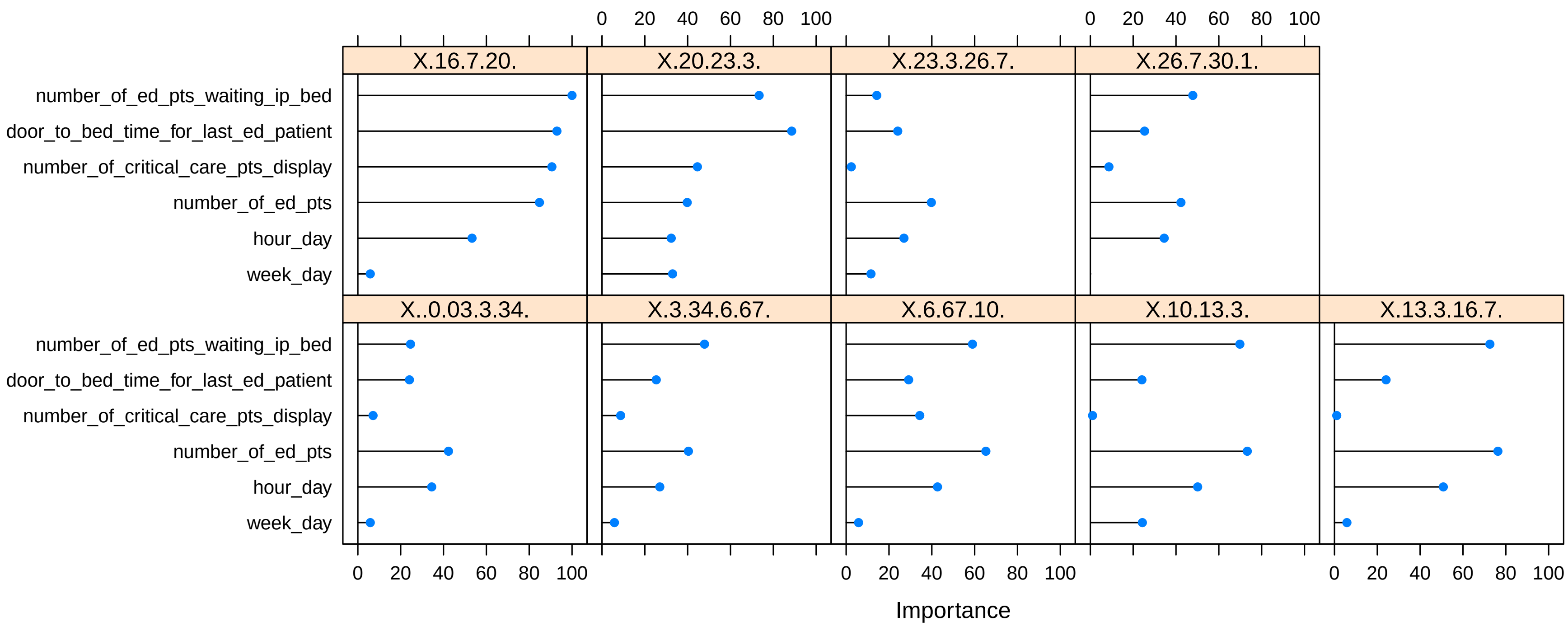

Figure 4: Variable Importance: Random Forest Model

4.3.3 *Naive Bayes Model.*

To run Naive Bayes model, I have divided the outcome variable into categories to create a categorical variable (wait_cat). The variable has nine categories. For instance, how many times did admissions delay for between zero and three minutes, and so on. This is the `wait_cat` variable (see section 3.4 on feature engineering).

I now run the Naive Bayes model and generate the variable importance plots (Edureka 2019). Figure 5 shows the variable imprortance by category. For instance, the top left plot shows the factors that are imprtant for patients that wait for between 20 and 23.3 minutes. In that case, the number of patients is critical, followed by the door to bed time for the last emergency department patient. For the other categories, we use the same interpretation.

Recommendation: The implication is that in days and hours where there is projected to be more patients, we should have additional staff and portable emergency wards and beds.

```r
## Create a hyperparameter tuning grid
Grid <- data.frame(usekernel = FALSE, laplace = 0, adjust = 1)
## Run the Naive Bayes model
mdl <- train(wait_cat ~ .,data = final_data,
  method = "naive_bayes", trControl = trainControl(method = "none"),
  tuneGrid = Grid
)
## Plot the variable importance
plot(varImp(mdl))
```

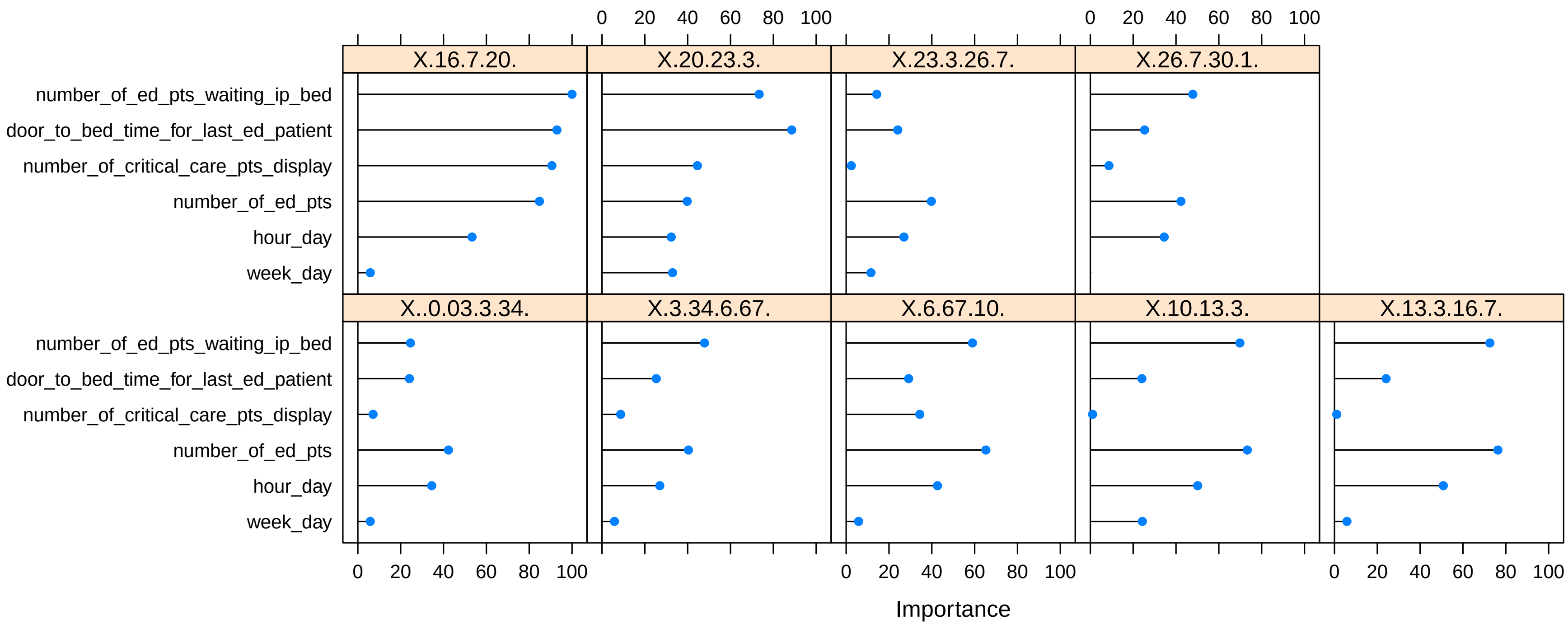

Figure 5: Variable Importance: Naive Bayes

4.3.4 *Random Forest Model.*

I run the random forest model (Muller and Guido 2016).

```
## Set random seed for reproducibility
set.seed(123)

## Run the random forest model
rf_model <- randomForest(longest_admit_time_waiting_in_ed ~ .
- wait_cat, data = waiting)

summary(rf_model)
```

```
##                 Length Class  Mode
## call              3    -none- call
## type              1    -none- character
## predicted       744    -none- numeric
## mse             500    -none- numeric
## rsq             500    -none- numeric
## oob.times       744    -none- numeric
## importance        6    -none- numeric
## importanceSD      0    -none- NULL
## localImportance   0    -none- NULL
## proximity         0    -none- NULL
## ntree             1    -none- numeric
## mtry              1    -none- numeric
## forest           11    -none- list
## coefs             0    -none- NULL
## y               744    -none- numeric
## test              0    -none- NULL
## inbag             0    -none- NULL
## terms             3    terms  call
```

For inference and actionable insights, I get the variable importance. Figure 4 below shows that hour of the day is of primary importance in explaining waiting times for patients. The number of patients and the number of patients waiting in inpatient beds are also important.

```r
## Variable importance plots for the random forest model

vip(rf_model) +
  labs(title = "Variable Importance: Random Forest Model")
```

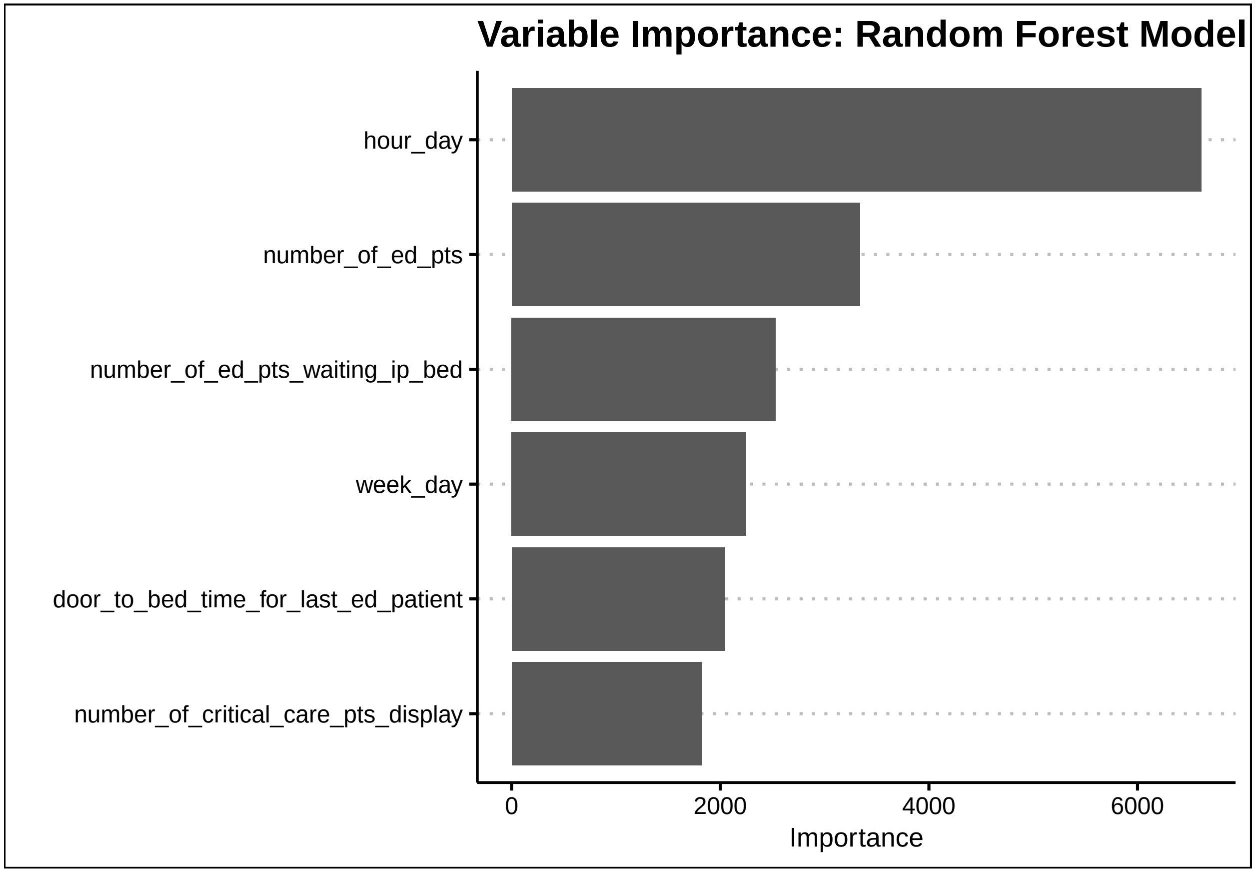

Figure 6: Variable Importance: Random Forest Model

Recommendation: Vary the number of staff by the hour of day where there are more emergency cases.

5 Key Takeaways

The four models tell us the critical factors associated with hospital delays. The regression model,for instance, points to the number_of_ed_patients_waiting_in_ip_beds, Weekday, and hour of day. Likewise,the SVM model also points to number of ed patients and hour of day as important drivers of long delays (for instance a delay of over 25 minutes). The observation is the same for naive bayes and random forest models. The models inform the need to manage staff, more so when there are many patients during particular hours of the day and days of the week. For forecasting purposes, it would be useful to split the data into a training set and test set and select the best model among the four.

6 Conclusion

In this analysis, I have used emergency department data from a fictitious hospital to generate insights to reduce waiting times. Following my analysis, I come up with the following insights.

- The critical factor in wait times are the number of patients.
- The number of patients vary by number of day or day of week.

My recommendations are as follows;

- Vary the number of staff depending on days of week that have more emergency cases.
- Vary staff by hour of day, with more staff during hours with more emergency cases like 1100 hrs, and less during times when there are fewer cases like around midnight.
- Given that the hospital has fixed number of beds, there should be arrangements to create portable, temporary emergency wards and beds to cater for the upsurge in emergency cases duting particular days of week and hours of day.

7 Packages Used in the Analysis

I have utilised the following packages in R for the analysis.

```
sessionInfo()
```

```
## R version 4.2.2 Patched (2022-11-10 r83330)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Linux Mint 21.1
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.10.0
## LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.10.0
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8    LC_NUMERIC=C            LC_TIME=en_US.UTF-8
##  [4] LC_COLLATE=en_US.UTF-8  LC_MONETARY=sw_KE       LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=sw_KE          LC_NAME=C               LC_ADDRESS=C
## [10] LC_TELEPHONE=C          LC_MEASUREMENT=sw_KE    LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
```

```
##  [1] kernlab_0.9-32       caret_6.0-94         lattice_0.20-45
##  [4] e1071_1.7-13         naivebayes_0.9.7     stargazer_5.2.3
##  [7] vip_0.3.2            randomForest_4.7-1.1 ggthemes_4.2.4
## [10] corrplot_0.92        GGally_2.1.2         conflicted_1.2.0
## [13] readxl_1.4.2         kableExtra_1.3.4     skimr_2.1.5
## [16] janitor_2.2.0        lubridate_1.9.2      forcats_1.0.0
## [19] stringr_1.5.0        dplyr_1.1.2          purrr_1.0.1
## [22] readr_2.1.4          tidyr_1.3.0          tibble_3.2.1
## [25] ggplot2_3.4.2        tidyverse_2.0.0      pacman_0.5.1
##
## loaded via a namespace (and not attached):
##  [1] colorspace_2.1-0     class_7.3-21         snakecase_0.11.0
##  [4] base64enc_0.1-3      rstudioapi_0.14      proxy_0.4-27
##  [7] listenv_0.9.0        farver_2.1.1         prodlim_2023.03.31
## [10] fansi_1.0.4          xml2_1.3.3           codetools_0.2-19
## [13] splines_4.2.2        cachem_1.0.7         knitr_1.42
## [16] jsonlite_1.8.4       pROC_1.18.0          compiler_4.2.2
## [19] httr_1.4.5           Matrix_1.5-4         fastmap_1.1.1
## [22] cli_3.6.1            htmltools_0.5.5      tools_4.2.2
## [25] gtable_0.3.3         glue_1.6.2           reshape2_1.4.4
## [28] Rcpp_1.0.10          cellranger_1.1.0     vctrs_0.6.2
## [31] svglite_2.1.1        nlme_3.1-162         iterators_1.0.14
## [34] timeDate_4022.108    gower_1.0.1          xfun_0.39
## [37] globals_0.16.2       rvest_1.0.3          timechange_0.2.0
## [40] lifecycle_1.0.3      future_1.32.0        MASS_7.3-59
## [43] scales_1.2.1         ipred_0.9-14         hms_1.1.3
## [46] parallel_4.2.2       RColorBrewer_1.1-3   yaml_2.3.7
## [49] memoise_2.0.1        gridExtra_2.3        rpart_4.1.19
## [52] reshape_0.8.9        stringi_1.7.12       highr_0.10
## [55] foreach_1.5.2        hardhat_1.3.0        lava_1.7.2.1
## [58] repr_1.1.6           rlang_1.1.0          pkgconfig_2.0.3
## [61] systemfonts_1.0.4    evaluate_0.20        recipes_1.0.5
## [64] labeling_0.4.2       tidyselect_1.2.0     parallelly_1.35.0
## [67] plyr_1.8.8           magrittr_2.0.3       R6_2.5.1
## [70] generics_0.1.3       pillar_1.9.0         withr_2.5.0
## [73] survival_3.5-5       nnet_7.3-18          future.apply_1.10.0
## [76] utf8_1.2.3           tzdb_0.3.0           rmarkdown_2.21
## [79] grid_4.2.2           data.table_1.14.8    ModelMetrics_1.2.2.2
## [82] digest_0.6.31        webshot_0.5.4        stats4_4.2.2
## [85] munsell_0.5.0        viridisLite_0.4.1
```

References

DataCamp. n.d. "Support Vector Machines in r - Tutorial." https://www.datacamp.com/tutorial/support-vector-machines-r.

Edureka. 2019. "Naive Bayes Algorithm in r: A Comprehensive Tutorial." https://www.edureka.co/blog/naive-bayes-in-r/.

Muller, Andreas C., and Sarah Guido. 2016. *Introduction to Machine Learning with Python.* O'Reilly Media. https://www.oreilly.com/library/view/introduction-to-machine/9781491976432/.