
CASE STUDY: MODEL ENGINEERING

University of Applied Science - Online

Master of Science Data Science

# Task 2: Automation of Standby Duty Planning for Rescue Drivers via a Forecasting Model

Mohammad-Marko Jibrini

Matriculation: 32103720

mohammadmarko.jibrini@iubh.de

# Contents

# I List of Figures



# II List of Tables

# 1 Introduction and Business Understanding

This task aims to forecast the number of standby rescue drivers for Berlin's Red Cross rescue service. The standby drivers are activated when the on-duty rescue drivers can't handle all the emergency calls. The current approach of having 90 rescue drivers on standby each day is deemed inefficient. On many days only a small percentage of standby drivers are activated. Furthermore, sometimes 90 standby drivers are insufficient to cover all the calls.

Therefore, this project aims to build a forecast model where the percentage of standby drivers that are activated is maximized while minimizing the number of days where there are not enough drivers to cover the calls. In other words, we want a high activation rate $\frac{nstandbys}{standbyneeded}$ and a high coverage rate (rate of days where there were enough standbys to cover all the calls) We are also interested in the precision of this model.

It is unclear how this project is funded and whether standby drivers are unpaid volunteers. But for funding and/or reporting purposes it could also be interesting to know the number of calls addressed by standby drivers.

# 2 Data understanding & preparation

## 2.1 Raw data understanding

We start by loading the data into a data frame object. We want to know the data type of each column and check for missing values. There are no missing values in this dataset. The dataset contains the following values 2.1 for each day between 1st of April 2016 and 27th of May 2019.Next we create histograms for each variable in 2.1

| Column name | Column Description | Dtype |
|---|---|---|
| date | entry date | Object |
| n_sick | number of drivers who called in sick | int64 |
| calls | number of emergency calls | float64 |
| n_duty | number of drivers on duty available | int64 |
| n_sby: | number of standby resources available | int64 |
| sby_need | number of standbys which are activated on a given day | float64 |
| dafted | number of additional drivers needed due to not enough standbys | float64 |

Table 2.1: Raw Data

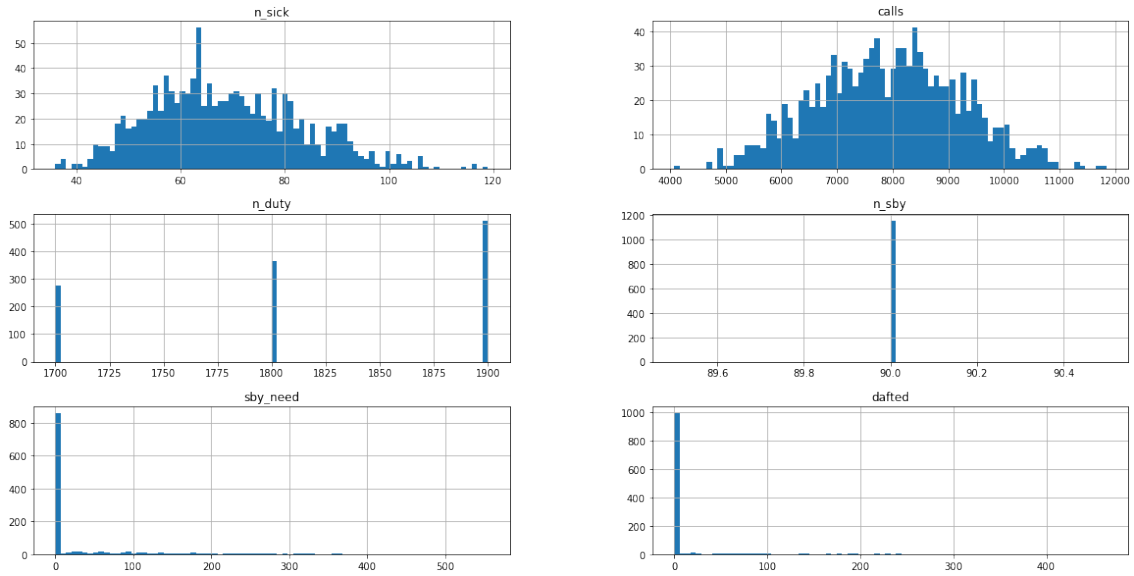



Figure 2.1: Histograms of all variables

As expected n_standby is a constant at 90. sby_need is most frequent at zero, revealing the number of days where no standby drivers were necessary. It has a maximum value of 555 on the first of April 2019. Similarly, dafted has an even stronger peak at 0. This will be all the days where the number of standby drivers needed is less than 90. Its distribution is the same as sby_needed shifted to the left by 90. As expected it has a maximum of 465 (555-90) on the first of April 2019. To investigate whether this is due to "April fools" prank calls, we look at calls in a range of days around the date in question. Figure 2.2 shows a particularly strong peak on April 1st, but it does not stand out from the general trend enough for it to be labelled an outlier.

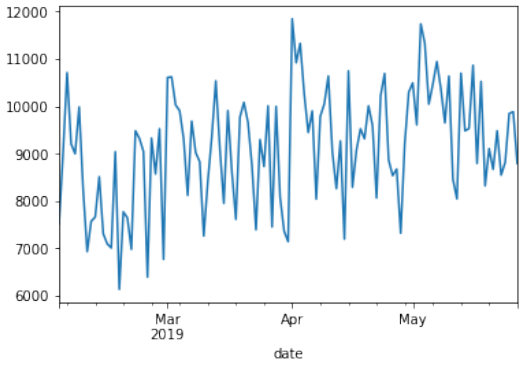



Figure 2.2: Calls around April 1st 2019

Next we move to the n_duty column. We can see in 2.1 that there are three possible values for the amount of on-duty drivers. To see what is going on, we plot the number of drivers against the Date in 2.3. The number of drivers starts at 1700, then rises by 100 at the start of 2017 and rises again at the start of 2018.

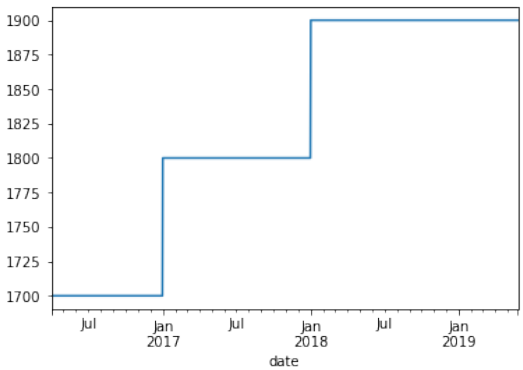
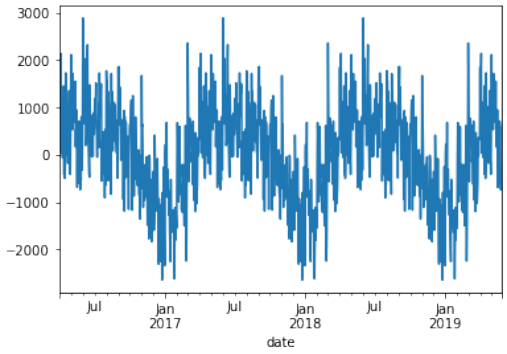
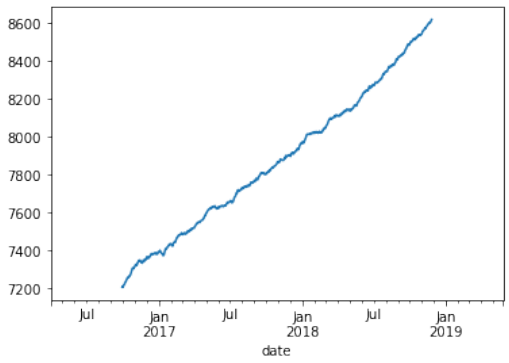



Figure 2.3: Number of drivers

Finally, we look at the seasonality and trend of the calls. Figure 2.4a shows the seasonal pattern of calls. The calls reach a low during the winter and peak during the summer. Figure 2.4b shows the trend, which is steadily and linearly increasing with time. The slope is around 1.5 meaning on average there will be an increase of 1.5 calls each day.




(a) Seasonality of calls




(b) Trend of calls

## 2.2 Target Variables

The target variable of this project is the amount of standby drivers needed each day. Instead of trying to predict it directly, it makes more sense to predict the number of calls and the number of drivers that call in sick. Then we look at the relationships between these and the number of standby drivers needed. For this to work we are making two assumptions:

1. The total number of drivers can be deduced from the total number of calls
2. Standby drivers are as effective at dealing with the calls as on-duty drivers, or there is a linear relationship between the two.

To test the validity of our assumptions, we first need to add a new column of total drivers to our data. To do that, we add the number of drivers on duty and the number of standby drivers needed and subtract the number of sick drivers. Then we create a new column of calls per driver by dividing the number of calls by the total number of drivers. Note that here we only use days where the number of standby drivers needed was larger than 0 but smaller than 90. Figure 2.5 shows the distribution of our newly created variable. We can deduce that each driver can deal with exactly 5 calls per shift (the minor deviation is explained by rounding). Furthermore, we can deduce that standby drivers are exactly as effective as on-duty drivers. If this was not the case we would expect a larger deviation (calls per driver would be lower on the days when more standby drivers were activated). Therefore, both of our assumptions are justified. The number of standby drivers needed can be determined by:

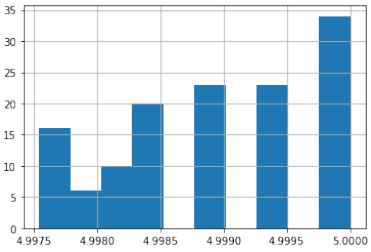



Figure 2.5: Calls per Driver

$$\frac{calls}{5} - n\_duty + n\_sick = sby\_need \tag{2.1}$$

## 2.3 Feature engineering

Now that we have a solid understanding of our data, especially the relationship between our input and target variables, we can move on to feature engineering. First we change the "date" column to a DateTime type and set it as the index. Next we extract the year, month, day and weekday information from the date. Now we test the claim: "during winter months more rescue drivers call in sick". Here we look at the percentage of drivers that call in sick, as the number of on-duty drivers is not constant

in our data. Figure 2.6 shows that there is indeed an increase in the number of drivers who call in sick during September and October.

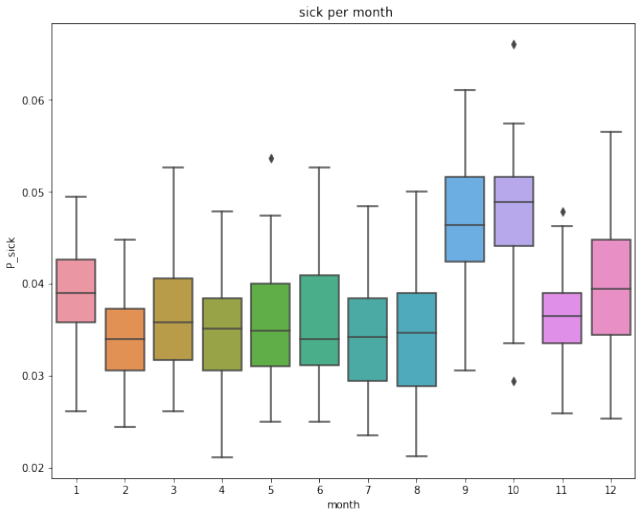



Figure 2.6: Percentage of sick drivers each month

The whole process is broken down into 3 steps:

1. Predict the number of de-trend calls and the percentage of sick drivers from date information
2. Re-trend the calls data and find the total number of sick drivers
3. Predict the number of standby drivers needed.

# 3 Modeling

## 3.1 Evaluation

Before we start building the model, we need to define evaluation metrics better. From the task: "efficient means that the percentage of standbys being activated is higher than in the current approach of keeping 90 drivers on hold. It also means that situations with not enough standbys should occur less often than in the current approach". We will use the coverage rate to determine the rate of days where enough standby drivers were present. For the percentage of standby drivers being activated we could look at the percentage of standbys being activated each day then find the mean, or we could look for the total percentage of standbys activated for the whole time period that we are evaluating. The latter is a more suitable metric as the former would not discriminate between large errors where no standby drivers were needed. We also add three additional factors: " Mean absolute error", "Average number of shifts" and " Calls addressed by standby drivers". The dataset is very volatile and the number of standbys needed varies a lot during the year. For each model we will run the evaluation on the whole dataset and a validation dataset to give us a better idea of its performance. Full results can be found in 4.1

For the current model of having 90 standby drivers each day, the evaluation is- Coverage rate: 0.85, Average activation rate: 0.20, Mean absolute error: 88, Average shifts per day: 90.0, Answered calls by standby drivers per day: 92.

## 3.2 Baseline Model

For a simple baseline model, we first calculate a pivot table of calls and the percentage of drivers sick. Then we set the predicted number of calls and percentage of sick drivers to be the corresponding entry from that pivot table. So we are only taking into account the month and weekday for our predictions. For consistency, we will only use data up to the end of March 2019 for training this model. Figure 3.1 shows the number of calls predicted by this model. While the baseline model captures the average and trends of the data, it fails to identify the peak numbers of calls. This is problematic as it is those peaks that are associated with needing standby drivers. Therefore, the average shifts per day according to this model is just over 3. To address those issues we can shift the model in a way that can specify the average number of shifts we want it to produce. Figure 3.2b represents the adjusted model, where the average shifts per day are set to 90. Now the metrics for the whole dataset are: Coverage rate: 0.9, Average activation rate: 0.28, Mean absolute error: 75, Average shifts per day: 91, Answered calls by standby drivers per day: 127.

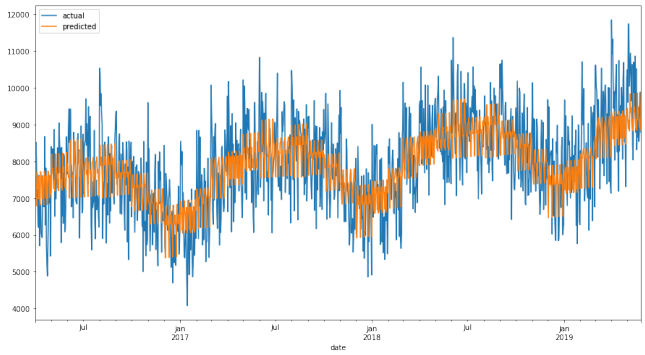
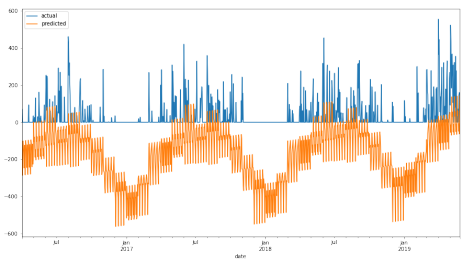
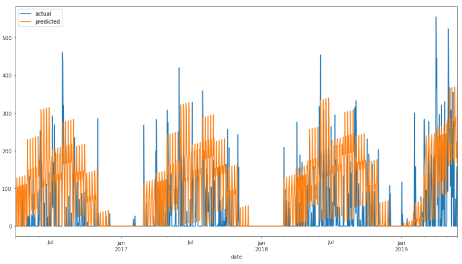



Figure 3.1: Baseline Prediction of number of calls



(a) Baseline raw sby needed



(b) Baseline adjusted sby needed

## 3.3 Predictive Model

Linear regression, support vector regression and XGboost were tested for this model. With and without dummy variables and with a combination of date time info. XGboost regressor performed slightly better than the rest.

After choosing the model, the next question was whether or not to use lag features. Using the number of calls over the last n number of days to help predict the number of calls tomorrow. Training with lag

features did improve the precision of the model. However, when predicting multiple days in advance, the lag features themselves become a prediction. Given that this model will have to be run on the 15th of the current month to predict the shifts for the next month, the benefit from using lag features is lost. Without the lag features this problem becomes a simple regression task with date information as input and predicted standbys needed as output.

While building the baseline model we learned that it is the peaks of the number of calls that matter the most, as they determine when standbys are needed. In order to address this we can over-represent dates where such peaks exist by duplicating those rows. Days when the calls needed more than 90% of drivers on duty were dupicated 5 times. Rare days where the number of standbys needed was over 200 were duplicated 15 times.

After a bit of parameter tuning, we got two XGboost regressors. One for de-trended calls and another for the percentage of sick drivers. Combining them we get the following results for the whole dataset: Rate of days where not enough standbys were present: 0.201, Average activation rate: 0.494, Mean absolute error: 35.16, Average shifts per day: 35, Answered calls by standby drivers per day: 86.65 This is the most precise model so far. With a high activation rate. Compared to the current model, the amount of calls addressed by standby drivers has decreased by less than 5%, while decreasing the number of shifts by more than 60%. If the goal was to be efficient with standby drivers, we would stop here. The main problem with this model is that it underperforms in the rate of days where not enough standby were present compared to the baseline model. It can be argued that this is the most important metric and therefore, this issue should be addressed.

We create another "max" model. It is a copy of the previous model except the number of calls predicted is increased by a factor x such that $calls\_max = calls * x$. Where x is chosen so that the rate of days where not enough standbys were present is 0.05. To get the final prediction we use $calls\_final = calls + n * (calls\_max - calls)$. We can plot n against coverage or the number of shifts 3.3. We can also access the value of n that corresponds to a particular coverage rate or a particular number of shifts.

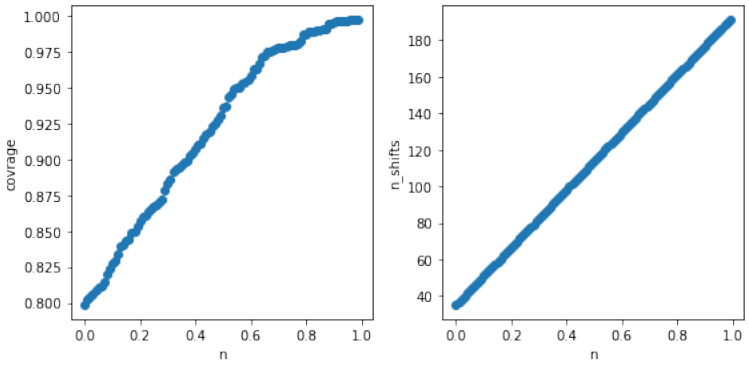



Figure 3.3: n factor

# 4 Results

The table 4.1 summarizes the findings. All models have been trained on data from 01/04/2016 to 31/03/2019. The table shows results for the whole data and the validation data that was not included in the traning. Note that the validation data is the most recent one and thus, due to the increasing trend in calls received and it being in peak time, it has a larger number of standbys needed than the rest of the dataset. This leads to lower coverage numbers. Model n=0.35 was chosen as it corresponds to the average shifts being 90. Note that here 90 shifts is the average for the date range present. For future models it will predict more than 90 average standbys needed according to the trend in calls. As

the average increase in calls is approximately 1.5 a day, the average increase in total drivers needed per year is $\frac{1.5*365}{5} \approx 110$. To keep the number of standby drivers constant each year the number of on-duty drivers should increase by $\frac{110}{1-p} \approx 114$ where p is the average probability that a driver will call in sick.

| | dates: 01/04/2016-27/05/2019 | | | | dates: 15/04/2019-27/05/2019 | | | |
|---|---|---|---|---|---|---|---|---|
| Model | Covrage | activation | n shifts | error | Covrage | activation | n shifts | error |
| (n=0) | 0.8 | 0.49 | 35 | 35 | 0.57 | 0.63 | 109 | 103 |
| (n=0.35) | 0.9 | 0.32 | 90 | 67 | 0.67 | 0.52 | 195 | 125 |
| (n=1) | 0.999 | 0.18 | 193 | 158 | 0.98 | 0.37 | 355 | 224 |
| baseline | 0.9 | 0.28 | 90 | 75 | 0.74 | 0.48 | 224 | 147 |
| Current | 0.85 | 0.2 | 90 | 88 | 0.51 | 0.59 | 90 | 120 |

Table 4.1: Results

Figure 4.1 shows the predictions of different models in the validation data.

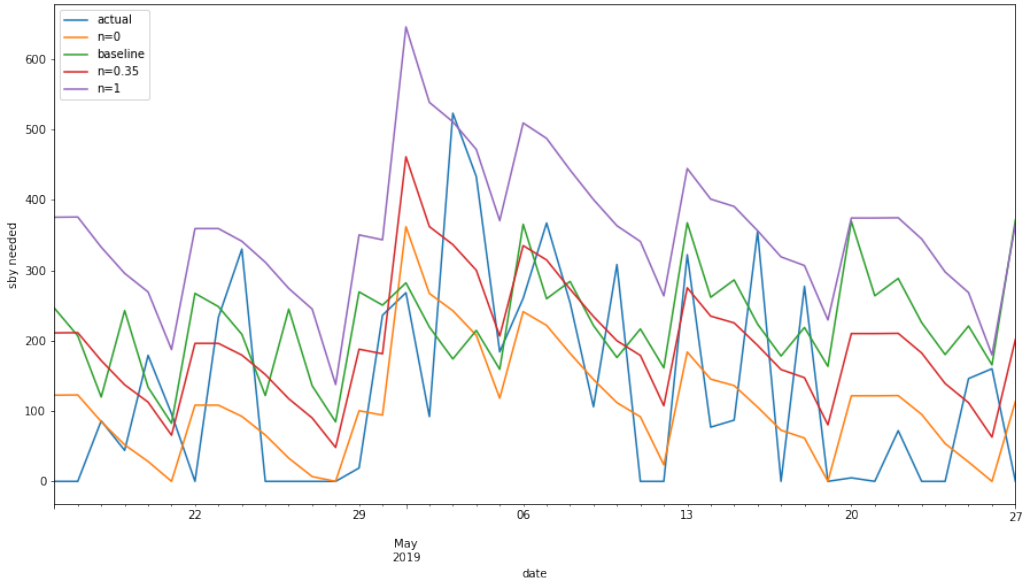
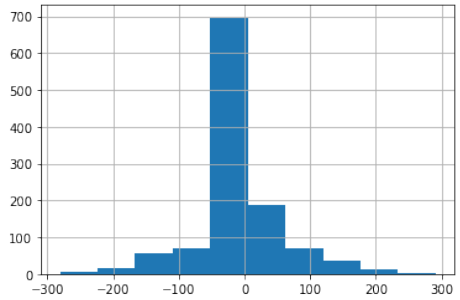



Figure 4.1: Predictions for May 2019

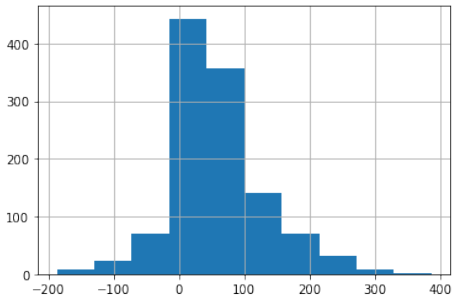
## 4.1 Errors

As can be seen from table 4.1, model n=0 is by far the most precise, but it underperforms when it comes to coverage. We can trade off precision for coverage by increasing n. For this error analysis we will use the n=0.35 model. Figures 4.2a and 4.2b show the difference in the error distribution between the two models.There is an increase at 0 due to all the predictions where no standbys were needed being set to 0.

Figure 4.3 shows the error percentages for the n=0.35 model. There is 50% chance that the error will be smaller than 54, and 75% chance that the error is smaller than 95. The error in predicting the number of standbys needed is fully due to the errors of our predictions on the number of calls and the number of sick drivers. The error in the number of calls is the dominant factor. When we take out all the entries where standbys needed = 0, the correlation between the errors on calls and standby drivers is over 0.98. As mentioned previously, we can reduce this by reducing n. By increasing n we are purposefully overestimating the number of calls leading to larger errors.




(a) Error mode n=0




(b) Error model n=0.35

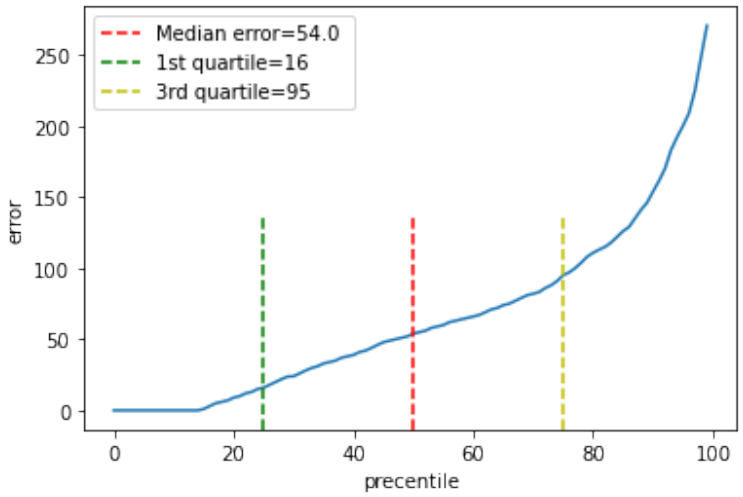



Figure 4.3: Error percentiles (e=0.35)

# 5 Conclusions

For this project we have created a model to predict the number of standby drivers needed. Instead of giving a single prediction, we allow our users to make a value decision between precision and coverage.

I propose GUI to be presented to the final user where they could input a date range for which they want the predictions. A slider by which the user can control the factor n. Users should have the option to choose n to be presented as coverage or the number of shifts. In each case, they could choose their preferred metric, and then the model will choose the suitable n before returning the results.

The model should be re-trained on new data each month. There is plenty of room for improvement by trying different models and during further parameter tuning. Factor x, which determines the "max" model, should be revisited periodically.