# INF264
# Project 2:
# Predicting traffic

Tellev Sundt (tsu012) and Johanna Jøsang (fak006)

October 9, 2020

NOTE: for a better view of the graphs that we included in this pdf, please see the jupyter notebook we handed in. Most of the text in this pdf are repeated next the to appropriate codeblocks, so you can follow the notebook without missing important information.

## 1 Preprocessing

### 1.1 Visualization

We imported the dataset using pandas and saved it in a dataframe. To get a general idea of what the data looked like we started by printing it and looking at what features there were and what type of values the features had.

| | År | Måned | Dag | Fra_time | Volum til SNTR | Volum til DNP | Volum totalt |
|---|---|---|---|---|---|---|---|
| 0 | 2015 | 12 | 16 | 11 | 265 | 232 | 497 |
| 1 | 2015 | 12 | 16 | 12 | 243 | 223 | 466 |
| 2 | 2015 | 12 | 16 | 13 | 251 | 289 | 540 |
| 3 | 2015 | 12 | 16 | 14 | 369 | 409 | 778 |
| 4 | 2015 | 12 | 16 | 15 | 283 | 365 | 648 |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 35187 | 2019 | 12 | 31 | 19 | 77 | 72 | 149 |
| 35188 | 2019 | 12 | 31 | 20 | 58 | 61 | 119 |
| 35189 | 2019 | 12 | 31 | 21 | 52 | 51 | 103 |
| 35190 | 2019 | 12 | 31 | 22 | 51 | 39 | 90 |
| 35191 | 2019 | 12 | 31 | 23 | 34 | 81 | 115 |

[35192 rows x 7 columns]

Figure 1: Simple print of the dataframe

Thereafter we wanted to get a first general impression on how to the total traffic volume varied over the years, so we plotted all the data in a graph with a long x-axis.

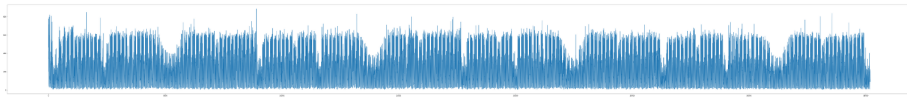

Figure 2: All the data plotted

While it was possible to see the cyclical nature of the data, better visualization was required to properly identify trends in the data.

After splitting the data by year, and plotting the data for each year individually we could see that the shape of the data for each year was quite similar. Unfortunately we did not have a full year of data for 2015, but looking at 2016-2019 was enough to see the similarity in each year.

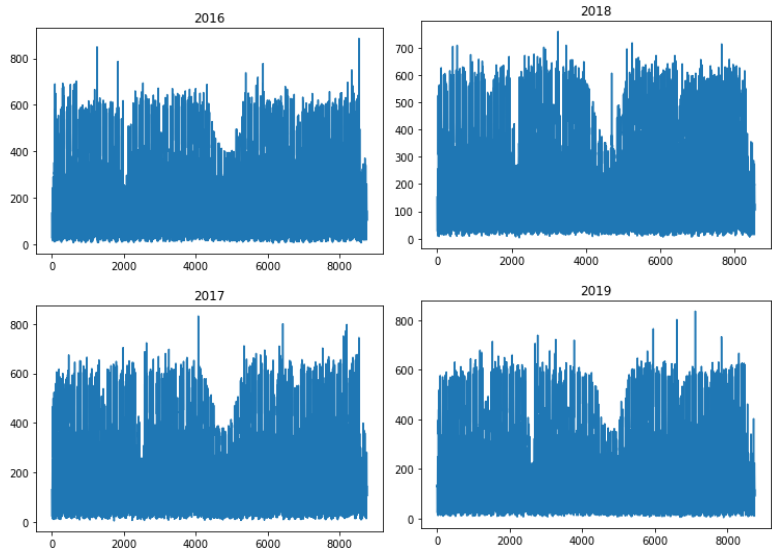


Figure 3: Data for years 2016-2019

Up until now we had simply plotted one data point after the other, so the x-axis merely represented the index for each data point. This doesn't give us enough insight into the data, so we changed the x-axis to represent time intervals.

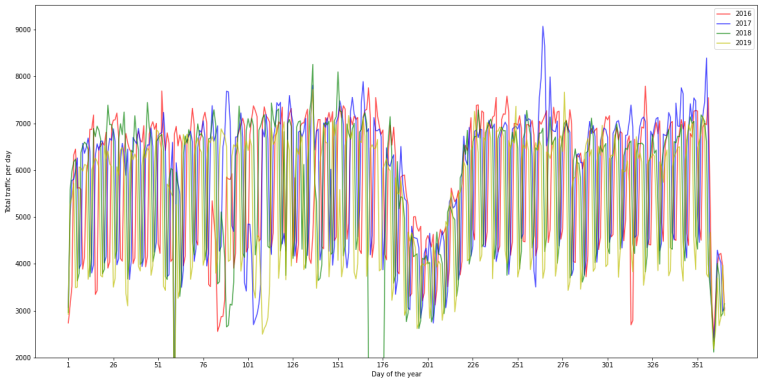


Figure 4: Total traffic per day

Plotting the total traffic per day of the year for years 2016-2019 revealed a consistent trend of the traffic falling much lower every weekend, and the dip in traffic over the summer and Christmas holidays was visible. It was however still quite distracting the have the constant fluctuation between week-day traffic and week-end traffic, so we made a new plot with total traffic per week.

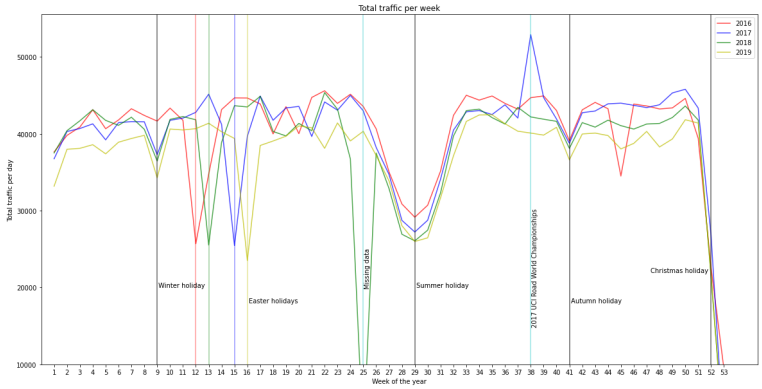


Figure 5: Total traffic per week

Now the similarity is traffic for each year was a lot easier to identify, and it was possible to clearly mark holidays on the graph. Being content with our understanding of the large-scale cycles in the data, we then took a look at the cycles that happen in a day. We plotted the mean total traffic per day of the week, in addition to the mean of the in- and out-traffic from the city centre.

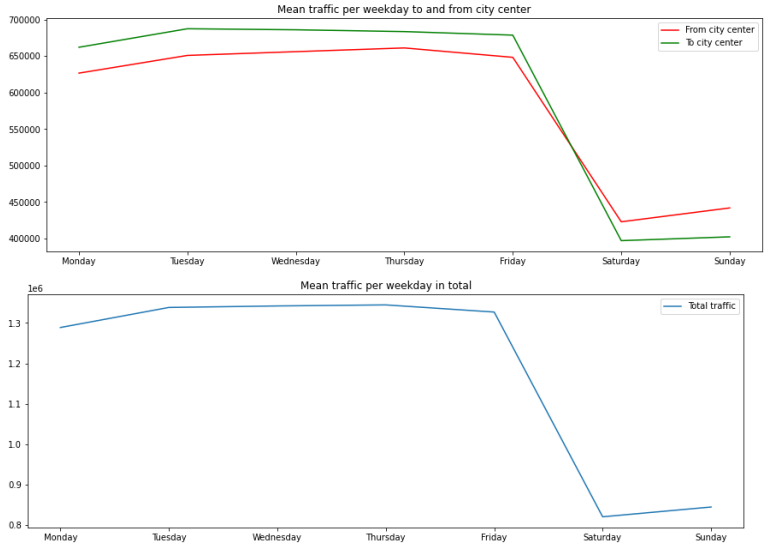


Figure 6: Mean traffic per weekday

From this we could confirm what was assumed, namely a considerable drop in traffic on the weekends, and otherwise a relatively consistent level of traffic in the weekdays. Finally we wanted to look at the traffic throughout the day, so we plotted the mean traffic per hour of the day.

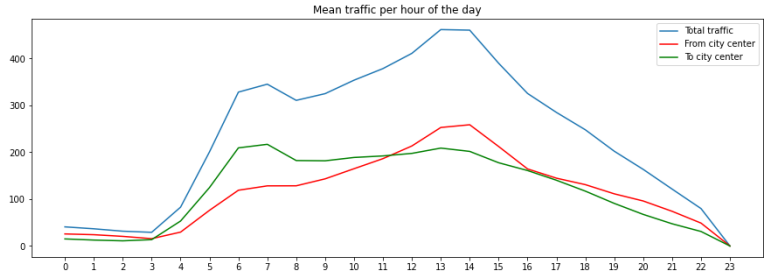


Figure 7: Mean traffic per hour of the day

This looked as expected, with traffic to the city center peaking in the morning and traffic from the city center peaking in the afternoon. The total volume of traffic peaks a bit past, and the traffic volume slowed down to very low numbers in the middle of the night. The second largest peak in traffic is in the morning, which we could speculate to be because most jobs in Bergen are in the city center.

## 1.2 Feature engineering

### 1.2.1 Holidays

We know that during holidays there is less traffic, and we could see this in the data too. Therefore we decided to add an "is holiday" feature, which was 1 for a datapoint that was considerered to be in a holiday, and 0 otherwise. Unfortunately we did not come up with an elegant way of finding the holiday intervals, so we manually wrote down the weeks and single days that we considered to be holidays in Bergen.

### 1.2.2 Cyclical nature of the traffic

We want to convey the cyclic nature of the hours of the day, and unfortunately the way the hours are represented in the original data gives no indication that the 23rd hour is close to the 0th hour. To solve this we will use two triginometric functions:

$$\sin\left(\frac{2\pi t}{24}\right) \quad \text{and} \quad \cos\left(\frac{2\pi t}{24}\right)$$

where t is the hour of the day. We add two features one with sin and the other with cos, because if we only used one of the two, then two hours in the day would have the exact same value, indicating a relationship where there is none. We do the same with days of the year, where $d$ representes the day number in the year:

$$\sin\left(\frac{2\pi d}{367}\right) \quad \text{and} \quad \cos\left(\frac{2\pi d}{367}\right)$$

### 1.2.3 Daylight saving time

Since daylight saving time (DST) shifts traffic with one hour, we decided to add the feature "is DST". The feature has value 1 if it is daylight saving time, and 0 otherwise.

# 2 Model selection

Some of the models took quite a bit of time to run, so checking many different hyperparameters was very time consuming. If we had more time to work on this project, we would have checked more hyperparameters before selecting a model.

## 2.1 Cross validation

We wanted to use cross validation to get an as unbiased estimator as possible, but since we are using time series data a normal k-fold cross validation may not provide good results. The model is going to be used to predict traffic levels in the future, so by taking a chunk of the data for testing we end up using future data to possibly predict past data. In order to avoid this, we use a new method

for cross validation which always uses future data for testing. The split method we used is sklearns TimeSeriesSplit. It starts by taking a small partition in the beginning of the dataset for training, and then using a section of the data following for testing. In the next iteration the previous testing data is included in the training, and the next set of data is used for testing.

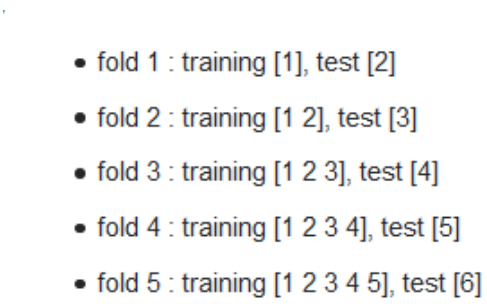


Figure 8: Visualization of TimeSeriesSplit cross validation procedure. Source

We found that using a 4-fold cross-validation worked well, but if we had more time we would have iterated for more values to check this. We chose 4 because splitting the data into four sections ensures that the training set always was large enough to contain data from parts of the year that the test data has. If we used a larger k, then in the first iteration the training data may have only consisted of the final months of 2015. When the testdata then containes data from the first months for 2016, the model needs to predict traffic on a time of the year it has never been trained on before.

## 2.2 Accuracy measure

Unlike MSE and RMSE, MAE is not as sensitive to large errors. Therefore we deemed MAE to be the most appropriate performance estimator, as there will always be new phenomena, such as a major sporting event, that can cause the model to make large errors in its prediction. For cross-validation we therefore take the average of the MAE for each iteration, in order to get a more unbiased estimator of the performance of the models.

## 2.3 Baseline

Before we started evaluating some models, we constructed a naive model to have as a baseline. We used sklearns linear regression and the cross validation method described above.

| Prediction | MAE |
|---|---|
| Total traffic | 68.621 |
| Traffic to SNTR | 38.889 |
| Traffic to DNP | 34.579 |

Table 1: Average MAE for linear regression models

### 2.4 Random forest regressor

We quite liked the decision trees from the previous project, so when we found random forest regressors, which corrects the decision tree's habit of overfitting, we were eager to use it. While there are a number of hyperparameters, we decided that the most important one was allowed tree depth. Therefore we iterated over many different depths, finding the best one for our case.

| Prediction | Optimal depth | MAE |
|---|---|---|
| Total traffic | 12 | 30.469 |
| Traffic to SNTR | 12 | 16.161 |
| Traffic to DNP | 10 | 18.342 |

Table 2: Average MAE for random forest regressor models

### 2.5 Regression tree

We checked the same parameter with regression trees, iterating over different depths to find the optimal one. For both random forest regressor and regression tree we found through experimentation that after a depth of 20 the performance of the model only worsened. Therefore we iterated over all the depths between 1 and 20 to find the best one. It may of course be possible that what we encountered was a local minima, and the true optimal depth for the trees is a much higher number.

| Prediction | Optimal depth | MAE |
|---|---|---|
| Total traffic | 9 | 33.847 |
| Traffic to SNTR | 8 | 17.777 |
| Traffic to DNP | 9 | 20.228 |

Table 3: Average MAE for regression tree models

### 2.6 Neural networks

We experimented a bit with the number of layers in the neural network, but didn't notice much difference in the performance of the predictor. So instead

we decided to go for one layer, which seemed to be enough, and instead iterated over the number of neurons in that layer.

| Prediction | No. neurons | MAE |
|---|---|---|
| Total traffic | 450 | 0.043 |
| Traffic to SNTR | 450 | 0.036 |
| Traffic to DNP | 450 | 0.038 |

Table 4: Average coefficient of determination for neural network models

#### 2.6.1 Model selection

Since MAE isn't appropriate for comparison between the normalized data used for the neural networks model, and the traffic to- and from the city which is half that of the total volume, we decided to look at the coefficient of determination to confirm our belief that random forest regressor was the best model.

| Model | Coef. of det. |
|---|---|
| Random forest | 0.915 |
| Regression tree | 0.887 |
| Neural networks | 0.885 |

Table 5: Average coefficient of determination for models predicting TOTAL traffic volume

# 3 Model Testing

Random forest regressor proved to be the best model. We use 10% of the data for model testing. This 10% corresponds to the last few months of 2019, so in this case the models have not been tested on data from across the year. With more data we could have tested the models on an entire year and gotten av more accurate performance estimate.

| Prediction | MAE |
|---|---|
| Total traffic | 25.595 |
| Traffic to SNTR | 13.072 |
| Traffic to DNP | 17.075 |

Table 6: Average MAE for selected models on unseen testing data

# 4 Use of model on 2020 traffic

On the data from 2020 our model perfomed relatively well, up until the lockdown started in march. It did quite badly then, which is expected, as the traffic usually is much higher during that time of the year. As the summer started and Covid-19 measures were eased the traffic went back to normal, and our model starts predicting relatively accurately again. From this we can conclude that while

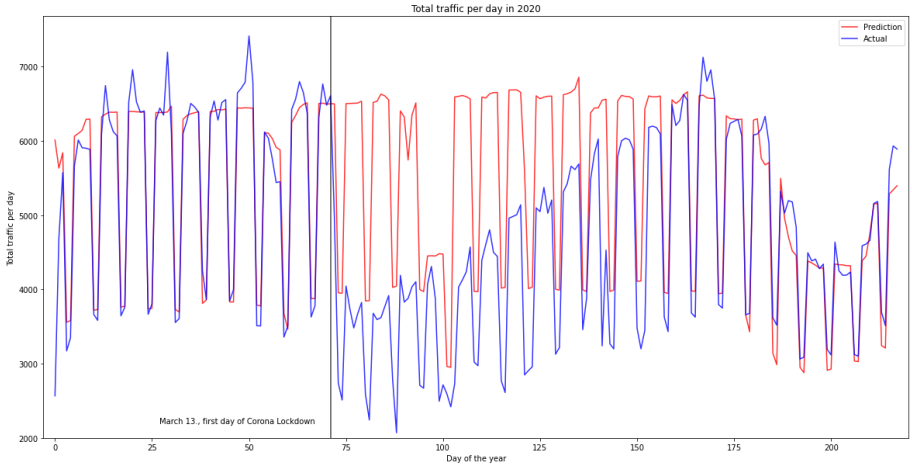


Figure 9: Predicted traffic and true traffic of 2020

our model is good at predicting traffic under "normal" circumstances, there may always be new phenomena, such as big sporting events or a pandemic, which it is not possible for our model to predict. Therefore it is important to be aware of the restrictions of the model, and understand that it really has a very limited representation of what causes traffic trends.