Universität St.Gallen

School of Management, Economics, Law,
Social Sciences, International Affairs and Computer Science

Data Science Fundamentals Project Report
Fundraising at the Bahnhofstrasse Zurich

Data Science Fundamentals I 8,580,1.00

Fall Semester 2023

Teachers

Lyudmila Grigoryeva

Jonathan Chassot

Hannah Busshoff

Editors

Berit Schrader I 22-615-330

Anne Bally I 22-606-297

Johannes Aschoff I 22-616-528

Hans Richard Eden I 22-608-186

St.Gallen, 10.12.2023

Particulars

Berit Schrader I 22-615-330
Altmannweg 1, 9012 St.Gallen
+ 41 (0)76 205 53 72
berit.schrader@student.unisg.ch

Anne Bally I 22-606-297
Blumenbergplatz 5, 9000 St.Gallen
+41 (0)89 549 4198
anne.bally@student.unisg.ch

Johannes Aschoff I 22-616-528
Innerer Sonnenweg 5, St.Gallen
+41 (0)78 635 17 77
johannesmaximilian.aschoff@student.unisg.ch

Hans Richard Eden I 22-608-186
Guisanstrasse 69, 9010 St.Gallen
+41 (0)78 243 6324
hansrichard.eden@student.unisg.ch

Table of Contents

List of Figures ..... **III**

List of Tables ..... **IV**

1. Introduction **1**

2. Analysis and Imputation of the Target Variable **2**

2.1. Visualization of the Target Variable 3

2.2. Imputation der Target Variable 4

3. Analysis and Imputation of the Features **8**

3.1. Non-binary and Hourly Features 8

3.2. Non-binary and Non-hourly Features 10

3.3. Binary and Hourly Features 13

3.4. Binary and Non-hourly Features 13

3.5. Relationship Between the Hourly and Non-Binary Features and the Target 15

4. Preparation of the Variables for Modelling **16**

4.1. Log-Transformation of the Variables 16

4.2. De-seasoning of the Variables 17

4.3. Status After Preprocessing 20

5. Building the Model **21**

5.1. LSTM Recurrent Neural Network 21

5.2. Tree-Based Methods 25

6. Conclusion **28**

List of References ..... **V**

List of Figures

Figure 1: Overview of the Three Sectors of the Bahnhofstrasse 2
Figure 2: Overview of the Available Data for the Three Sectors of the Bahnhofstrasse 3
Figure 3: Logarithmic Distribution of the Target Variable 4
Figure 4: Results of Mean Imputation Illustrated Across Different Timeframes 5
Figure 5: Results of Factor Imputation Illustrated Across Different Timeframes 5
Figure 6: Results of Multiple Factor Imputation Illustrated Over Different Timeframes 6
Figure 7: Comparison of MSE-Results of the Different Imputation Methods 7
Figure 8: Comparison of the Imputations of the Best Four Imputation Methods 7
Figure 9: Visual Analysis of the «Traffic» Feature 9
Figure 10: Visual Analysis of the «Sunshine» Feature 10
Figure 11: Visual Analysis of the «Consumer Price Index» Feature 11
Figure 12: Visual Analysis of the «Retail Trade Turnover» Feature 11
Figure 13: Visual Analysis of the «Population» Feature 12
Figure 14: Visual Analysis of the «Hotel Guests» Feature 12
Figure 15: Visual Analysis of the «Weather» Feature 13
Figure 16: Correlation Matrix and Distributions of the Hourly Features and the Target 15
Figure 17: Distribution and Relationship Between Monthly Data 15
Figure 18: Visual Analysis of the Unmodified Data (Histogram, (P)ACF) 16
Figure 19: Visual Analysis of the Data After Applying the Logarithm (Histogram, (P)ACF) 17
Figure 20: Visual Analysis of the De-seasoned Data (Histogram, (P)ACF) 17
Figure 21: Visual Analysis of the LOESS De-seasoned Data (Histogram, (P)ACF) 18
Figure 22: Before-After Comparison of Taking the Logarithm and De-seasoning 19
Figure 23: Visualization of Persisting Weekly Seasonality 19
Figure 24: Coefficients of Lasso Regression of the Target on Our Features 22
Figure 25: Visualization of Prediction's Training and Validation MSE 23
Figure 26: Comparison of Predictions of the First LSTM and the Actual Traffic 23
Figure 27: 24h Sequence LSTM Training and Validation MSE 24
Figure 28: Comparison of Predictions of the 24h Sequence LSTM and the Actual Traffic 24
Figure 29: Optimal Cost-Complexity Parameter After Cross Validation 25
Figure 30: First Three Levels of the Decision Tree Regressor 26
Figure 31: Predictions on the Test Data Generated by the Decision Tree Regressor 26
Figure 32: Predictions on the Test Data Generated by the Random Forest 27

List of Tables

Table 1: Overview of the Four Categories of the Feature Variables..... 8

Table 2: Overview of Categorized Variables and the Respective Preprocessing Measures 20

1. Introduction

The Bahnhofstrasse in Zurich is arguably the most frequented shopping street in Switzerland with an astonishing CHF 2.7bn being spent there every year. Of course, this is a great opportunity for fundraising organizations to raise money for their cause. The only question: When should they do so? In the morning, during the lunch break or rather in the evening, when people leave their workplace? This is exactly where we jump in: By analyzing input variables like weather and traffic volume, we developed a model that may help to predict the times of the day at which the Bahnhofstrasse is the most frequented. Thus, with our help, fundraising organizations will no longer have to rely on their gut feeling but can make well-founded and – most importantly – data-driven decisions.

2. Analysis and Imputation of the Target Variable

Our target variable p_t is the pedestrian frequency at the Bahnhofstrasse. It is measured hourly in real-time by several sensors that have been installed at the Bahnhofstrasse since the 28th of September 2021. This resulted in an aggregation of approximately 18'500 data points until the beginning of November 2023. The collected data is stored in a CSV data frame on the Zurich Open Data Platform (Urban Deveopment, Presidential Department, 2023). The data on the pedestrians is split on two levels:

1. There are three zones, reflecting the left side, the right side and the middle of the street respectively.
2. There are three sectors, reflecting the northern (*bhfs_north*), southern (*bhfs_south*) and middle part (*bhfs_middle*) of the Bahnhofstrasse respectively. The three sectors are depicted in the illustration below.

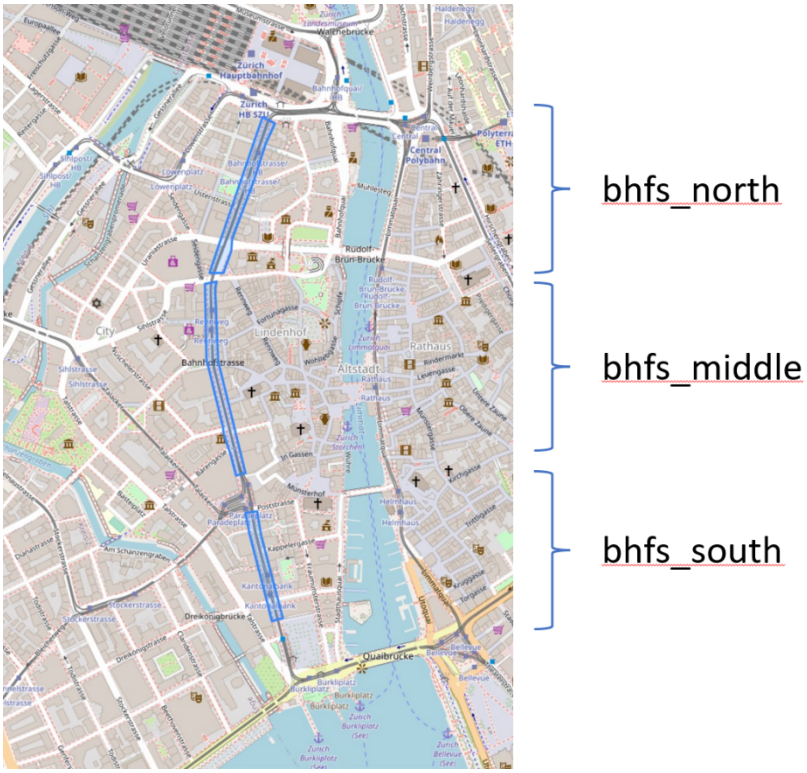


Figure 1: *Overview of the Three Sectors of the Bahnhofstrasse*

Because the three zones do not contain meaningful information for our purpose, we merged them. In contrast to that, we kept the separation between the three sectors of the Bahnhofstrasse for our analysis of the target p_t before merging them into one single target variable column.

2.1. Visualization of the Target Variable

When plotting the three different sectors, we noticed two major gaps in the data for *bhfs_north* that are also mentioned on Zurich's Open Data Platform. They amount to roughly one-third of the total *bhfs_north* data points[1]. Their cause lies in construction works around Zurich central station between mid-September 2022 and December 2022 as well as April 2023 and mid-September 2023. Apart from that, there are minor missing's that amount to a little more than 0.5% of all data points in total[2]. They can be traced back to malfunctions of the sensors.

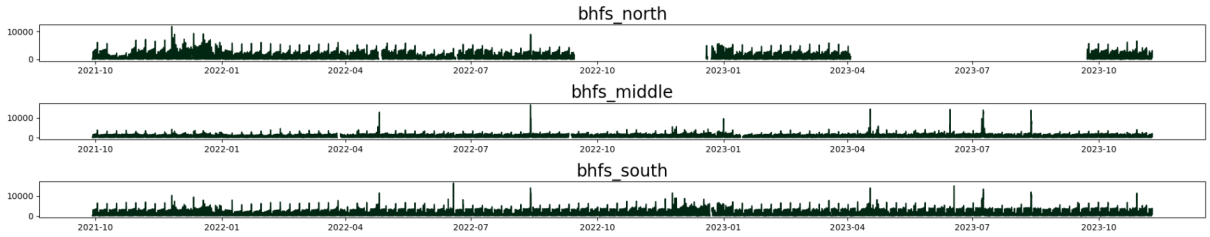


Figure 2: *Overview of the Available Data for the Three Sectors of the Bahnhofstrasse*

Another particularity that catches the eye when analyzing the plot is the seasonality of the data. In fact, before we visualized the data, we expected three different seasonal patterns in our data:

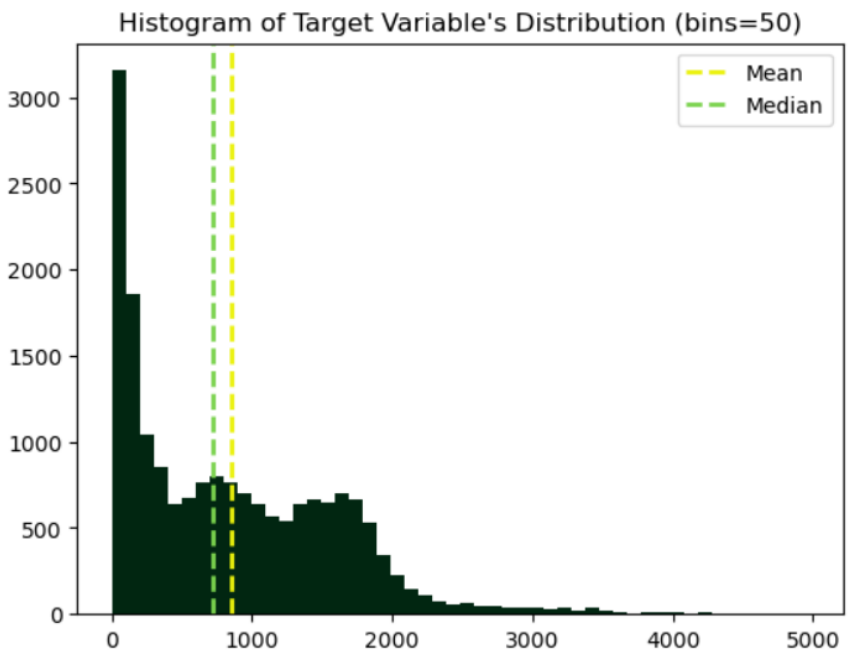
1. Daily Seasonality: For instance, there is more pedestrian traffic during the day than during the night.
2. Weekly Seasonality: For instance, the activity on the weekend differs from that during the week.
3. Yearly Seasonality: For instance, more people «live outside» during the summer than during the colder months of the year.

From *Figure 2*, we can conclude that the available period is too short to make an appropriate assessment of the yearly seasonality. By contrast, the weekly seasonality may be apparent. Regarding the daily seasonality, we cannot make a statement yet as the timeframe is too long to notice it. In *Chapter 4*, we delve deeper into our data's seasonality and find ways to deal with it.

Apart from the two major gaps of *bhfs_north* and the seasonality, we see that our data has a few outliers. A more detailed scrutinization confirms that *bhfs_middle*, for example, has a mean of around 900 while its highest value is above 16'000.

[1] 6515/18501

[2] 352/(18501*3)



The target variable may follow a logarithmic distribution rather than a Gaussian. It is heavily right-skewed, with its mode near zero and its mean and median around 850. Both, the outliers and the skewed distribution show a need for further processing, e.g. taking the logarithm, later on (see *Chapters 4* and *5*).

Figure 3: *Logarithmic Distribution of the Target Variable*

2.2. Imputation der Target Variable

Imputation of minor data gaps

We decided to impute the minor missings in between with the *Iterative Imputer* function from scikit-learn. To make it work best, we standardized our data with the function *StandardScaler*, as advised by scikit-learn (scikit-learn, n.d.a). The idea behind the decision to employ *Iterative Imputer* was to leverage the fact that we have our target variable split up into three different columns. As there may be a correlation between them, we could derive information from two columns to impute a missing in the third one.

As the *Iterative Imputer* is designed for multivariate datasets and can take multiple variables as input to find a corresponding value to impute the missings, it is sensical to employ it in our scenario. For instance, *Iterative Imputer* takes two inputs to impute a missing in *bhfs_middle*: *bhfs_north and bhfs_south*.

Imputation of the two major data gaps

After having imputed those minor missings in the three columns of our target variable, we went on to tackle the two major data gaps in *bhfs_north*. This time, we wanted to develop multiple solutions and find out which one minimizes the empirical risk, i.e. has the lowest Mean Squared Error (MSE) in our case. To be able to compare our different approaches, we first created a data frame without the timeframes where there is no data for *bhfs_north*. We then went on to split this newly created timeframe into a training and a testing subset[3] while

[3] With the training subset containing 2/3 and the testing subset containing 1/3 of the data.

maintaining the temporal order to avoid data leakage. With this preset, we could train each imputation method on the training subset to then compare its results on the testing subset with the actual entries for *bhfs_north*.

Mean Imputation

We started off by using mean imputation. However, we did not just calculate one mean across the board. Instead, we programmed a function that extracts individual means for every hour on every weekday from the existing *bhfs_north* data to have a more nuanced imputation.

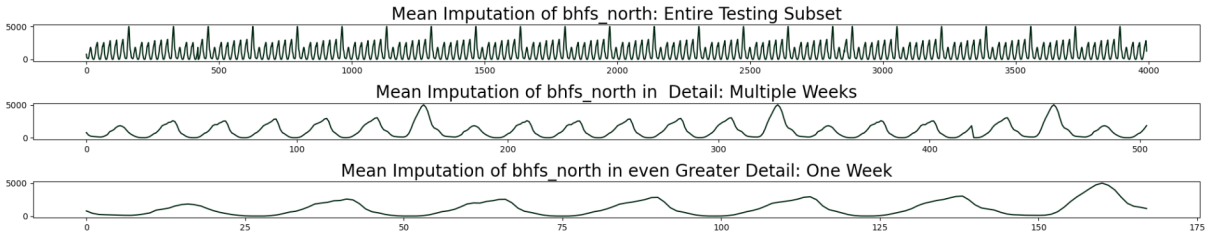


Figure 4: *Results of Mean Imputation Illustrated Across Different Timeframes*

As we can see in *Figure 4*, the function works well when only looking at the imputation for one week. When we zoom out, however, we notice that the imputation is too monotonous across multiple weeks and months.

Factor Imputation 1

Next, we tried out a «Factor Imputation». This means that we calculated one global factor for the mean of *bhfs_south* and *bhfs_middle* in relation to the mean of *bhfs_north* using the following equation:

factor = mean of bhfs_north / (mean of bhfs_south + mean of bhfs_middle)

This way, we leveraged the advantage of our target variable being split in three, as was laid out above. We then used the calculated factor to impute the two major gaps of *bhfs_north* based on the data that we have for *bhfs_south* and *bhfs_middle* for the respective timeframes.

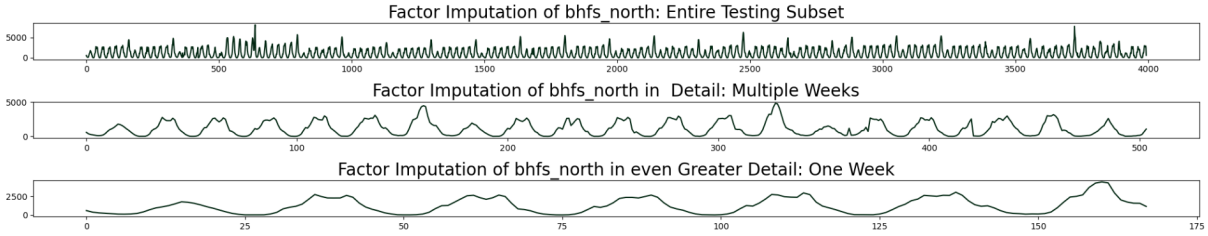


Figure 5: *Results of Factor Imputation Illustrated Across Different Timeframes*

Figure 5 illustrates how we now obtain a more balanced imputation when reviewing several weeks and months, compared to the mean imputation.

Factor Imputation 2

But there is an amelioration that can be made to achieve a result that may be even closer to reality: Instead of having a one-size-fits-all factor, we can calculate an individual factor for each hour on each weekday. Let us therefore call it «Multiple Factor Imputation».

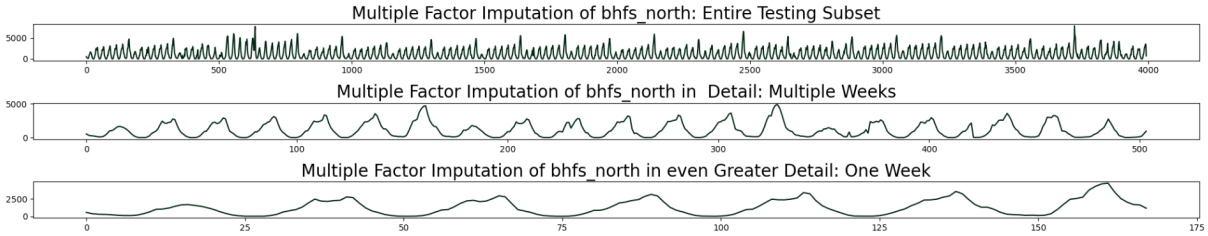


Figure 6: *Results of Multiple Factor Imputation Illustrated Over Different Timeframes*

Admittedly, the improvements compared to the simpler Factor Imputation above are barely visible when plotting the graphs. Nonetheless, the Multiple Factor Imputation works on a more granular level and thus may better reflect nuances in its imputation.

Further Imputation Methods

Apart from the three DIY-methods explained above, we also used three standard Python functions: *Linear Regression*, *Random Forests* and – as for the minor data gaps – *Iterative Imputer*, all provided by scikit-learn. This helped us to compare and thereby evaluate the effectiveness of our DIY functions.

This time, in contrast to the imputation of the minor gaps above, we were able to quantify the impact of preprocessing the data before employing *Iterative Imputer* by comparing the MSE's. However, we were not able to use standardization. This is because in order to inverse the standardization, say, of a column, one has to have standardized this column before, which is not the case in our scenario: In our testing subset, we only standardize the *bhfs_south* and *bhfs_middle* column to then populate the *bhfs_north* column with the *Iterative Imputer*. This means that we cannot inverse the standardization of the populated bhfs_north column. Thus, we substituted the standardization with logarithmizing which does not have the problem described above but still has a very comparable effect on the data.

The resulting MSE's show a noticeable positive effect of almost 20%[4] of preprocessing before employing *Iterative Imputer*. This confirms the course of action we took for the imputation of the minor data gaps with the *Iterative Imputer* (see above).

[4] (79112-64177)/79112

Comparison of Imputation Methods

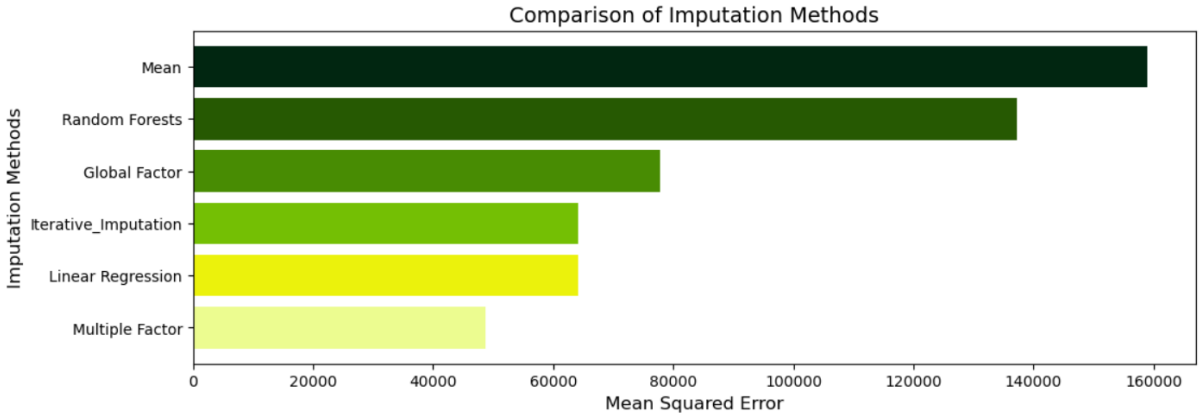


Figure 7: *Comparison of MSE-Results of the Different Imputation Methods*

Figure 7 illustrates the MSE's on the testing data of every imputation method we tried out. The Multiple Factor Imputation performs the best with a substantial advance of more than 35% compared to the Global Factor Imputation and an advance of almost 25% compared to Iterative Imputation and Linear Regression Imputation.

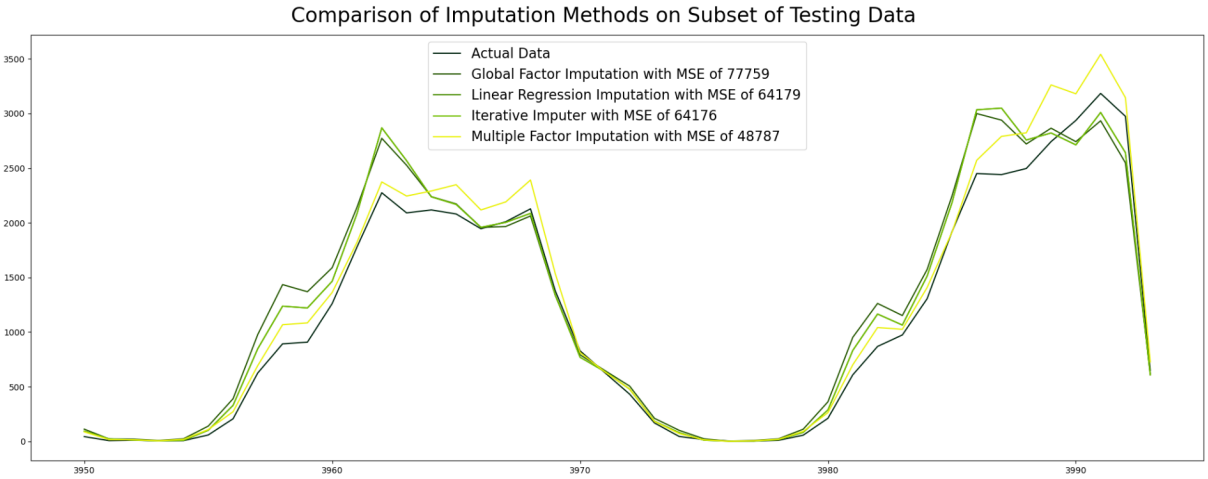


Figure 8: *Comparison of the Imputations of the Best Four Imputation Methods*

Figure 8 depicts how the Multiple Factor Imputation (light green line) sticks the closest to the actual data (dark green line). It manages especially well to retrace the actual pedestrian numbers for the data for the day. This is where every other imputation method has difficulties while all of them manage to capture the pedestrian numbers for the night quite accurately.

Given these results, we used the Multiple Factor Imputation to fill the two major gaps in *bhfs_north*.

After having scrutinized the target variable in detail, let us now move to our various feature variables.

3. Analysis and Imputation of the Features

In total, we collected eleven features that may have a relationship to our target variable, the number of pedestrians at a given time at Bahnhofstrasse. We categorized them into four different groups according to which this chapter is structured:

Table 1: *Overview of the Four Categories of the Feature Variables*

1. Non-binary and hourly features	2. Non-binary and non-hourly features
3. Binary and hourly features	4. Binary and non-hourly features

Finding hourly features was a challenge when we searched for potential candidates as there is not much data publicly available that is measured on an hourly basis. However, we needed them to reflect the hourly character of the target variable in our model. The binary features stem from dummy encoding, as we will describe in greater detail below.

The datasets were analyzed individually for the entire relevant timeframe to get a feeling for how to deal with the data subsequently. Furthermore, we imputed the missing values with individual methods and then merged them into a final data frame with a datetime column as an index.

On top of that, we shifted some of our features in time. This was necessary because in some cases, data only becomes available *after* the respective time that it was measured for. To avoid a look-ahead bias, we accounted for those instances in our model.

3.1. Non-binary and Hourly Features

Non-binary and hourly features include the traffic and the sunshine in the city of Zurich.

Traffic

The traffic data was taken from the «Stadt Zürich Open Data» website (Safety Department, Traffic Division, 2023). An official website of the government of Zurich based on the open government data initiative. The datasets display traffic count measurements for motorized individual traffic (MIT) in the city of Zurich since 2012. These values are provided daily and depict hourly values per traffic counting point. The data is collected at four different locations throughout the city. As all locations may have a relationship to our target, we took the sum of the four. The data is only available two days after being measured, which is why we had to shift all values two days in advance. Traffic was deemed potentially significant as it mirrors the vehicular movement within the city, which may coincide with the movements of pedestrians.

To properly analyze the distribution and course rate, we looked at the data on an hourly and daily basis. For the daily basis, we took the sum of all entries during the day.

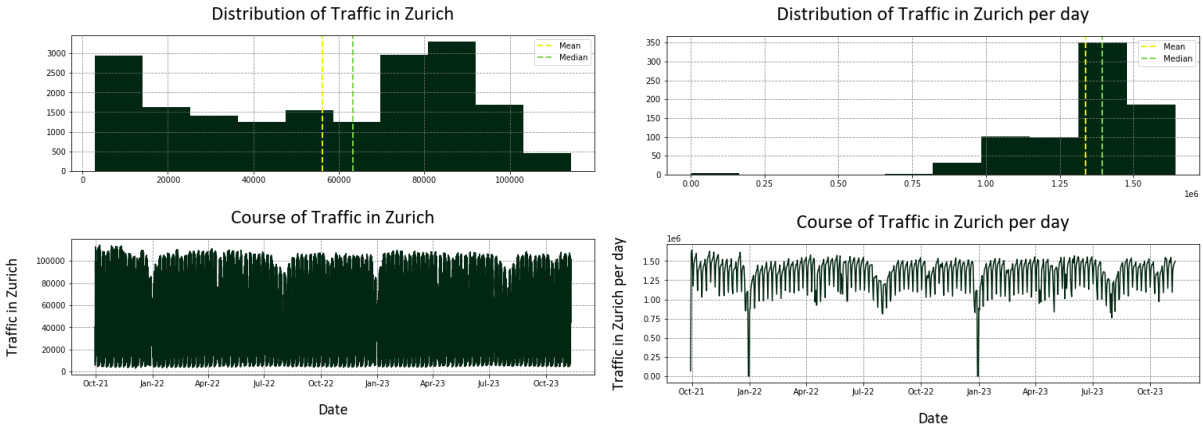


Figure 9: Visual Analysis of the «Traffic» Feature

The distribution of Zurich traffic on an hourly basis ranges from 2'936 to 11'4367 vehicles, has a mean of 5'6098 and a standard deviation of 3'1648 vehicles. When analyzing the course more closely, a seasonality can be detected, as there is a clear regularity in the behavior. How we took care of the seasonality will be discussed in *Chapter 4*. The distribution of the daily traffic in Zurich has a range of 0 vehicles a day to 1'641'400, a mean of 1'337'645 and a standard deviation of 201'885.

To ameliorate the analysis of the time series, missing values (e.g. due to a temporary outage of a counting point) have been replaced with imputed values. We observed that the data frame was missing values for entire days: They can be observed in the «Course of Traffic in Zurich per day» plot which shows two major spikes, attributable to the 30th – 31st December 2021 and 2022, respectively. Unfortunately, the website did not provide a reason for these gaps. Given the likelihood of higher-than-average traffic on these dates, we opted not to impute the missing values using a mean over the entire dataset. Instead, we chose to only utilize the mean of the previous day(s). To determine the optimal number of days for computing the mean, we assessed how the mean varies for different day ranges (1 to 9 days). The resulting average daily traffic for 2021 ranged from 998'581 to 1'166'400. Since the 29th of December is approximately on par with the average of these different day ranges, we employed the hourly data from this day of each year to impute the missing values for the entire days.

Sunshine

The sunshine data was taken from the «Stadt Zürich Open Data» website (Air quality measurement, environmental and health protection, Department of Health and the Environment, 2023). The data has been collected since 1992 by the Air Quality Measurement, environmental and Health Protection, Department of Health and the Environment. Importantly, during the ongoing year, the data is provisional and continually undergoes corrections. At the end of the year, the final data is published.

As the measured data is only available 30 minutes after the full hour, we had to shift all values one hour in advance. The data was collected on an hourly basis in four different locations in the city. We took the average of the sunshine across the different locations, as the overall

sunshine in the city may have a relationship with the pedestrian count and not just the sunshine at Bahnhofstrasse. We incorporated sunshine as it serves as a weather indicator. The weather may influence pedestrian outdoor activity.

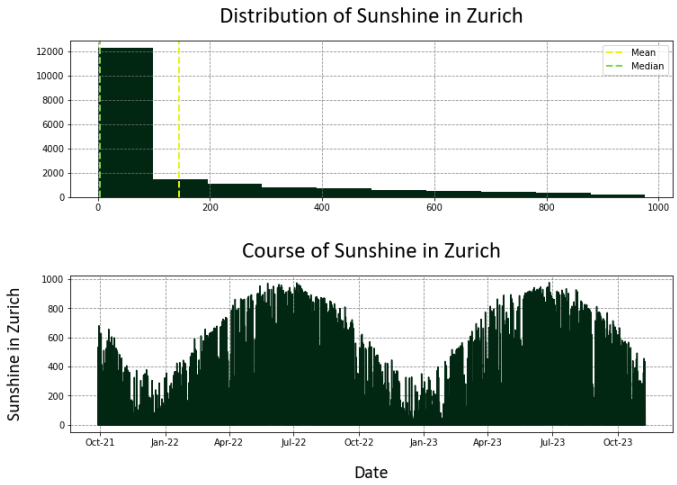


Figure 10: Visual Analysis of the «Sunshine» Feature

When looking at the course of sunshine, a clear seasonality is existent. How we handled the seasonality will be discussed in detail in *Chapter 4*.

Values for seven hours were missing in total. Because of the strong interdependence of the sunshine between subsequent hours, we imputed the missing values with the forward fill method, based on the values of the previous hours.

3.2. Non-binary and Non-hourly Features

Non-binary and non-hourly features include the Consumer Price Index, Retail Trade Turnover, population and hotel guests.

Consumer Price Index

The Consumer Price Index dataset was taken from the Federal Statistics Office (Federal Statistical Office, 2023), an official Swiss governmental website. The data was collected and is owned by the Federal Statistics Office of Switzerland. We incorporated the Consumer Price Index (CPI) because it tends to exhibit a pattern: an apparent linkage to decreased consumption of luxury goods during economic downturns and a corresponding rise in general goods consumption with inflation.

Originally, the dataset contained monthly data. To make the Consumer Price Index compliant with the pedestrian dataset, we added the values for the days and hours. This was achieved with the forward fill method, considering whether the data was available at the time to make a prediction. There were no missing values.

Below, *Figure 11* illustrates a histogram of the distribution of the CPI. The values ranges between 101.3 and 106.4. The mean lies at 104.3 and the mode at 104.6 with a standard deviation of 1.767. Furthermore, the graph displays the course of the CPI over time. It is apparent that the CPI is following a steadily increasing trend.

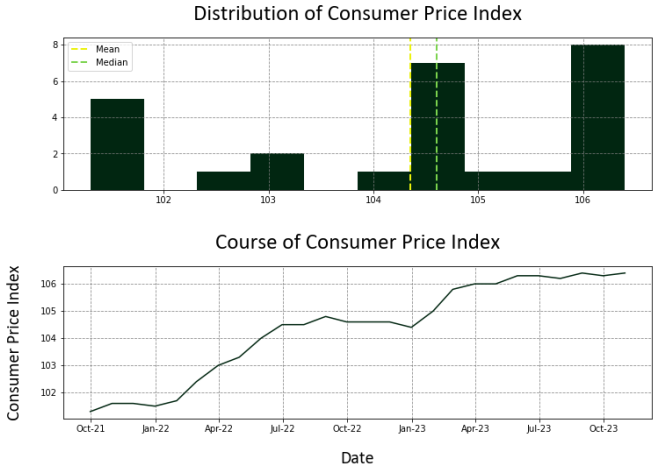


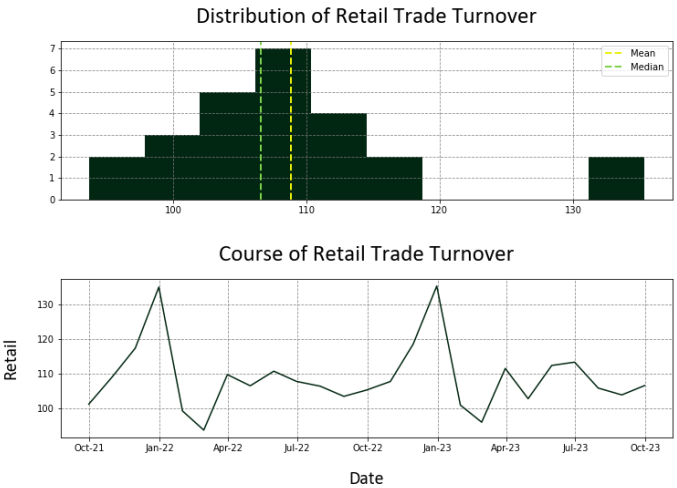
Figure 11: Visual Analysis of the «Consumer Price Index» Feature

Retail Trade Turnover

The Retail Trade Turnover dataset was taken from the Federal Statistics Office, an official Swiss governmental website (Federal Statistical Office, 2023). The rate is based on a national level and not just that of the city of Zurich. It is collected and owned by the Federal Statistics Office of Switzerland.

The Retail Trade Turnover may serve as a useful indicator for assessing the frequency of shopping visits in all of Switzerland. People from various neighboring cantons come to Bahnhofstrasse, reflecting the overall level of economic activity and consumer engagement. Observing fluctuations in turnover may have a relationship with the overall pedestrian frequency at Bahnhofstrasse.

As the CPI dataset above, the Retail trade Turnover dataset was originally structured monthly, which is why, with the forward fill method, we added the values for the days and hours. Furthermore, we shifted all values one month in advance, as the measured data is only available at the beginning of the next month.



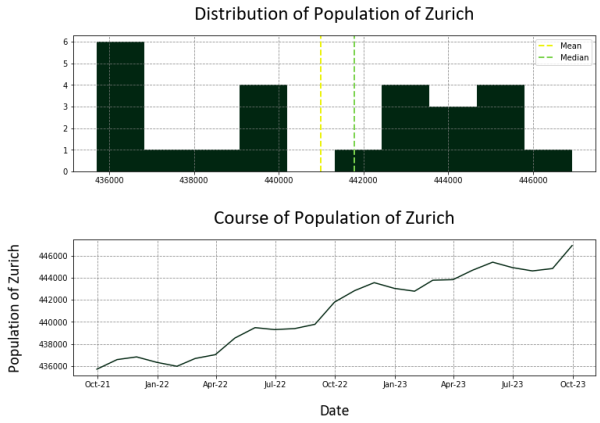
The distribution of the Retail Trade Turnover rate ranges between 93.7 and 135.4. It has a mean of 108.8 and a standard deviation of 9.9.

Figure 12: *Visual Analysis of the «Retail Trade Turnover» Feature*

Population

The population dataset of the city of Zurich was found on the «Stadt Zürich Open Data» website (Population Office, Presidential Department, 2023). The data has been collected every year since 1998 by the Zurich City Statistics, Presidential Department. It is owned by the Population Office, Presidential Department. During the ongoing year, the data is provisional and continually undergoes corrections. At the end of the year, the final data is published. As the measured data is only available at the start of the next month, we had to shift all values one month in advance. There were no missing values.

Over time, the population size of the city of Zurich may have a slight influence on the overall pedestrian frequency at the Bahnhofstrasse.

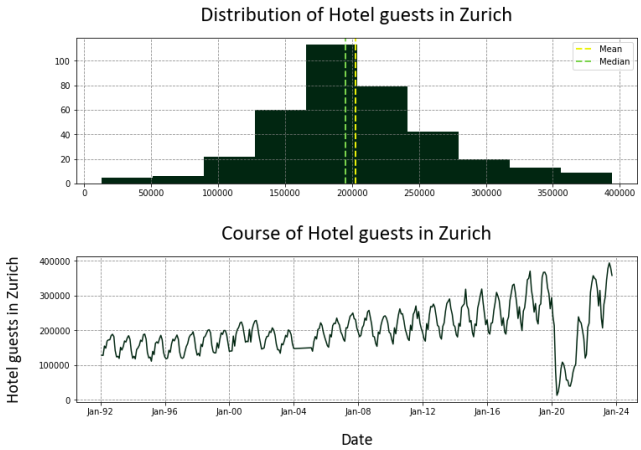


The distribution histogram indicates values between 435’712 and 446’920 inhabitants. A mean of 440’985 inhabitants and a standard deviation of 3’574. The population numbers were collected monthly, which is why with the forward fill method we added the values for the days and hours accordingly.

Figure 13: *Visual Analysis of the «Population» Feature*

Hotel Guests

The Zurich City hotel guest’s data was taken from the «Stadt Zürich Open Data» website (Zurich City Presidential Department, 2023). The Zurich City Statistics, Presidential Department has been collecting such data since 1992. It is owned by the Tourism Office, Presidential Department. During the ongoing year, the data is provisional and continually undergoes corrections. At the end of the year, the final data is published. The data includes all hotel guests in Zurich City, including children.



The measured data was only ever available at the start of the next month, which is why we shifted all values one month in advance. The count of hotel guests in Zurich city may offer an association with the footfall to areas like the Bahnhofstrasse, reflecting tourism and visitor activity within the city. The dataset did not contain any missing values.

Figure 14: *Visual Analysis of the «Hotel Guests» Feature*

The distribution histogram shows a mean of 202’799 guests with a standard deviation of 64’122. This is in a range between 13’053 and 394’143 guests. The dataset was structured monthly, which is why we added the values for the days and hours with the forward fill method.

3.3. Binary and Hourly Features

There is only one binary and hourly feature in our data frame: weather.

Weather

The weather feature was included in the original pedestrian data frame taken from the «Stadt Zürich Open Data» website (Urban Deveopment, Presidential Department, 2023). The data is owned by the Urban Development, Presidential Department.

The weather feature itself also did not differentiate between the different sections. It includes information on whether it was cloudy, foggy, raining, snowing, winding and – more generally – whether it was a clear-night, a partly-cloudy-day or a partly-cloudy-night. The individual categories were encoded with the *get_dummies* function. There were no missing values.

The weather condition that occurred the most was a partly-cloud-day and the condition that occurred the least was «windy», closely followed by «snow», as shown in *Figure 15.*

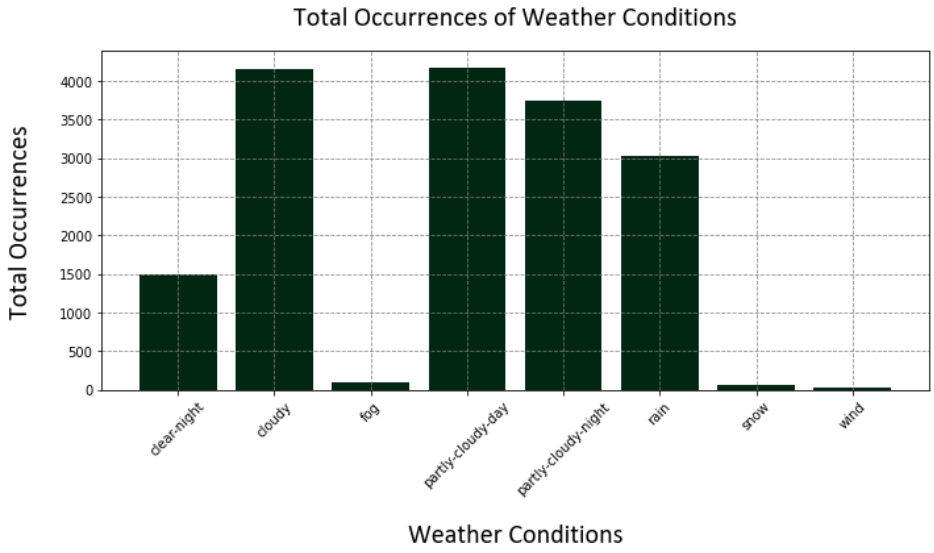


Figure 15: *Visual Analysis of the «Weather» Feature*

3.4. Binary and Non-hourly Features

Binary and non-hourly features include the features school holidays, closed stores and influential events.

School Holidays

The data was taken from the open holidays API (OpenHolidays API, 2023), a small open data project. It contains all school and public holidays for all European countries. Through the API we received the school holidays for each canton in Switzerland. We only took the holidays for

the neighboring cantons of Zurich and the canton of Zurich. The neighboring cantons include Schaffhausen, Thurgau, St. Gallen, Schwyz, Zug, and Aargau.

There may be an association between the school and public holidays and the people visiting the Bahnhofstrasse. Therefore, we created a data frame that indicates whether the canton of Zurich or a neighboring canton has school and/or public holidays. To do so, we inserted either a «1» if there was a school and/or public holiday on the respective day or a «0» if not.

Closed Stores

This dataset was created manually by taking the individual holiday dates from the official Zurich government website for the years 2021, 2022 & 2023 (Canton of Zurich, 2023). The dataset aims to indicate whether the stores at the Bahnhofstrasse were closed on the day, e.g. due to special holidays as we may assume there are fewer pedestrians at the Bahnhofstrasse then. We either assigned a «1» to the specific time and date if the stores were closed during that time or a «0» if not.

Influential Events

There are a few events every year, such as the marathon, that are responsible for thousands of people randomly passing Bahnhofstrasse. This dataset was also constructed manually by researching the dates of each big event that take place at or around the Bahnhofstrasse.

Most big events were found on the events webpage of Zurich (Zurich Tourism, 2023) and the rest by specifically searching for them. We assigned a «1» if such an event took place and a «0» if not.

3.5. Relationship Between the Hourly and Non-Binary Features and the Target

To better understand the connection between the hourly and non-binary features and the target, we computed the correlation of the monthly features with the target. Our findings, illustrated in *Figure 16*, reveal a positive linear relationship between the pedestrian count and both traffic and sunshine. It's important to keep in mind, however, that there may be non-linear relationships that are not fully captured by the Pearson correlation coefficient.

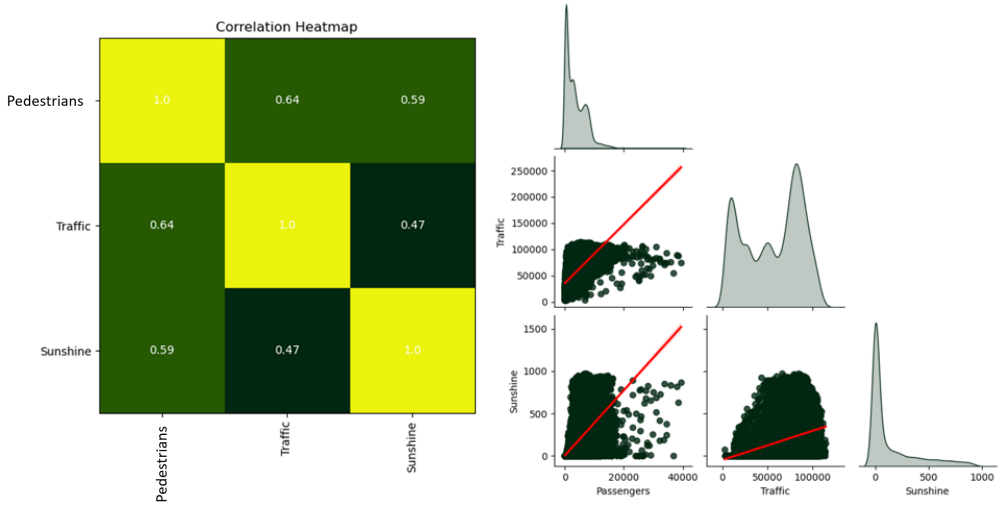
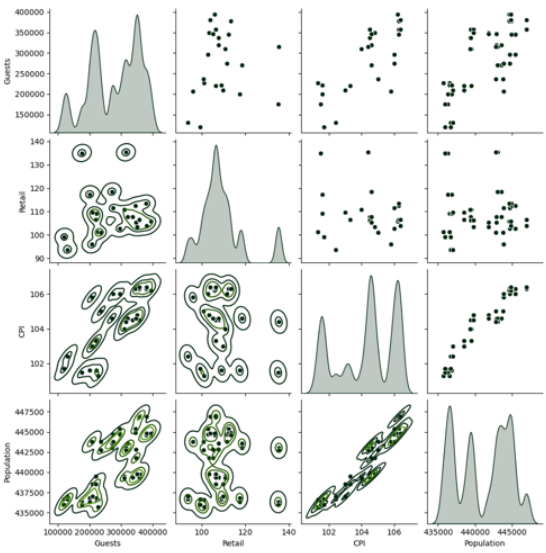


Figure 16: *Correlation Matrix and Distributions of the Hourly Features and the Target*



We also analyzed the relationship between the monthly features themselves. Results are shown in *Figure 17*.

As the number of available data points is very limited, we must be cautious to derive inference about potential relationships between the features. Yet, there may be a positive relationship ascertainable between the CPI and the Population as well as the number of Hotel Guests.

Figure 17: *Distribution and Relationship Between Monthly Data*

4. Preparation of the Variables for Modelling

To make a model generalizable, the target should be stationary (Jalil & Rao, 2019, p. 88). For example, the stationarity of the target is thus assumed in the ordinary least square model (Jalil & Rao, 2019, p. 87). A series y_t is stationary if (Mushtaq, 2011, pp. 2-3):

1. Its mean remains stable over time, $E(y_t) = \mu, \forall\, t$.
2. Its variance is time-invariant, $V(y_t) = \sigma^2, \forall\, t$.
3. The covariance of y_t and y_{t-s} is time-invariant but can be depended on the lag length, $Cov(y_t, y_{t-s}) = \lambda s$

To examine whether our target p_t is stationary, we plotted its distribution and the distribution of its rolling means over 24 days to see whether they are stable.

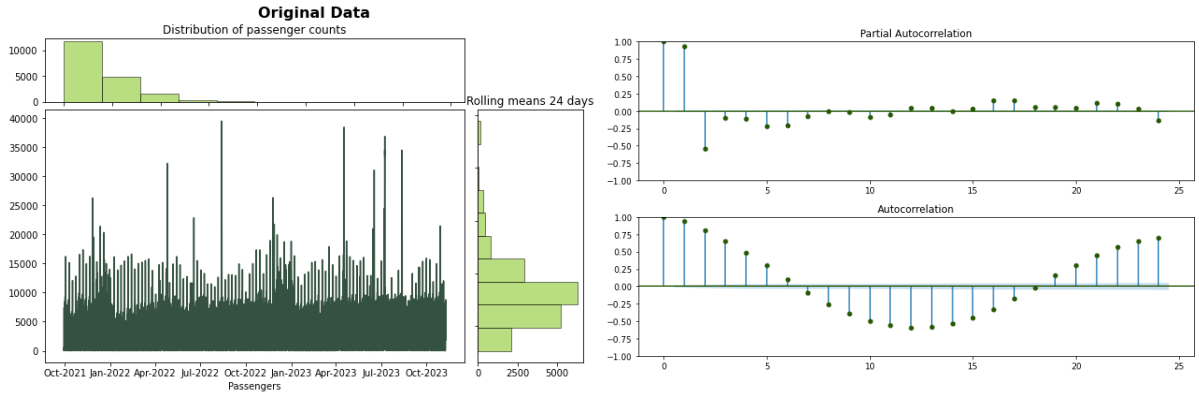


Figure 18: *Visual Analysis of the Unmodified Data (Histogram, (P)ACF)*

As evident in *Figure 18*, the distribution of the rolling means approximates a logarithmic distribution. The series does not seem to be stationary. To investigate our hypothesis further, we also examined the Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF) of p_t.

The ACF shows a clear pattern of seasonality, where the autocorrelation decreases until lag twelve and increases again until lag 24. This likely represents the daily season of the hourly pedestrian data. Before we began to de-season, however, we first had to take the logarithm of our variables.

4.1. Log-Transformation of the Variables

As we saw in the plots in *Chapter 2* and *3* that the distribution of our target variable and some features are skewed. Thus, we took the logarithm of the variables to stabilize their variance (Bisgaard, 2010, S. 72; Peixeiro, 2022, p. 47) and to linearize potential multiplicative trend and seasonal effects. In doing so, we can better meet the expectations of normality of the subsequent statistical tests.

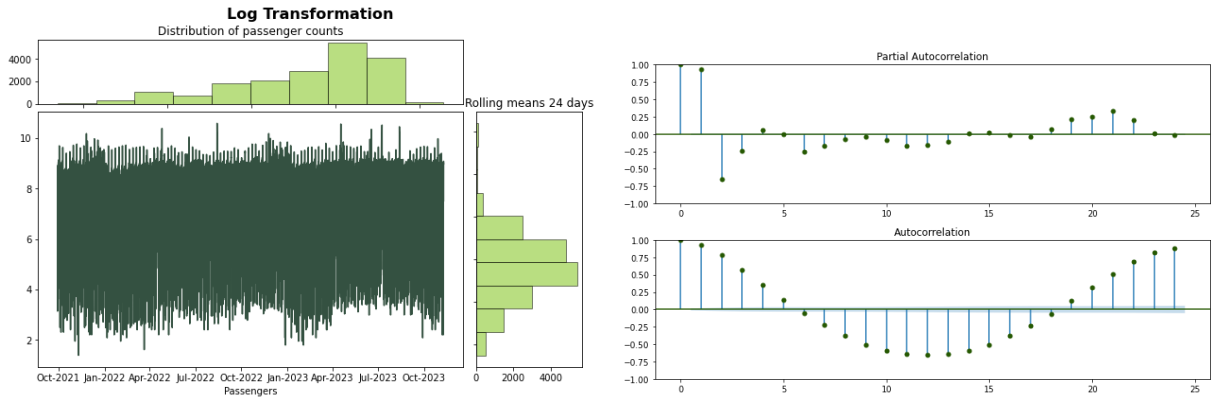


Figure 19: *Visual Analysis of the Data After Applying the Logarithm (Histogram, (P)ACF)*

The effect of applying the logarithm becomes evident when scrutinizing the histograms plotted in *Figure 19*. After taking the logarithm, they conform more closely to a Gaussian distribution.

4.2. De-seasoning of the Variables

After having applied the logarithm, we transformed p_t by taking its difference from p_{t-24} to eliminate the daily seasonality and stabilize the mean (Peixeiro, 2022, S. 47). Results are shown in *Figure 20*.

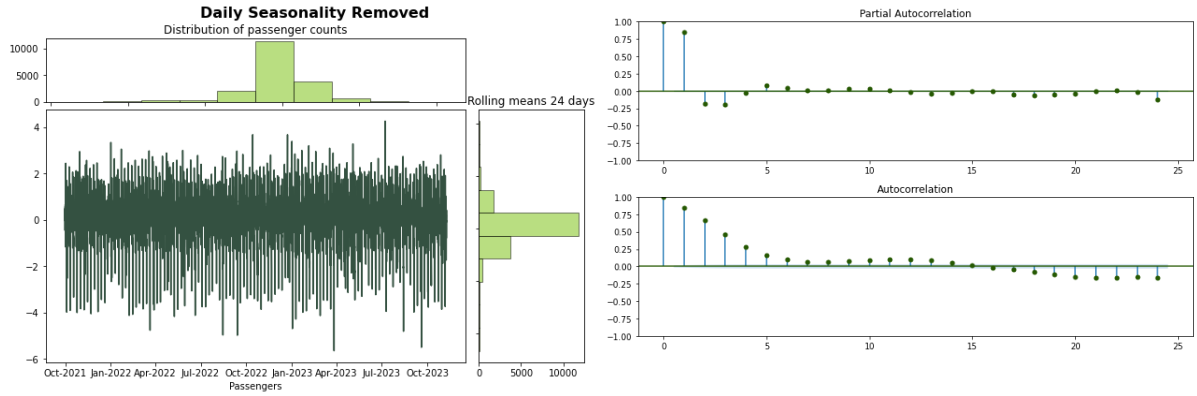


Figure 20: *Visual Analysis of the De-seasoned Data (Histogram, (P)ACF)*

While seasonality is still evident in the slowly decaying ACF, the rolling means are more stable. Splitting the series into three equivalent sizes yielded means of 0.00, -0.00, and -0.00 with standard deviations of 0.58, 0.59, and 0.62.

To test whether our transformation made p_t stationary, we employed the *Augmented Dickey-Fuller* (ADF) test and the *Kwiatkowski-Phillips-Schmidt-Shin* (KPSS) test.

The Dickey-Fuller test, developed by (Dickey & Fuller, 1979), has the following properties:

i. H_0: The series has a unit root (non-stationary)
ii. H_1: The series has no unit root (stationary)

We used the Augmented Dickey-Fuller test to allow for a higher-order autocorrelation as our ACF suggests a significant correlation even after 24 lags. The AIC was used to determine the

number of lags (statsmodels, 2023a). The result was a test statistic of -27.803 with a p-value of 0.00 suggesting that the null hypothesis that the series is non-stationary can be rejected.

To see whether the series is stationary around a deterministic trend (Kwiatkowski, Phillips, Schmidt, & Shin, 1992, S. 161), we have also used the KPSS test. It has the following properties (Kwiatkowski et al., 1992, p. 160):

i. H_0: The series has no unit root (stationary)
ii. H_1: The series has a unit root (non-stationary)

The test statistic amounted to 0.001 and the p-value exceeded 0.1 suggesting that p_t is stationary. However, as the ACF is still decaying only slowly, we also tried to eliminate the weekly trend by taking the difference of p_t with p_{t-7*24}. The results led to slightly lower test statistics for both tests and did not improve the ACF. Similar efforts to eliminate monthly and yearly trends also failed to enhance the ACF or test statistics. While the presence of trends such as Christmas shopping in December was expected, they did not manifest prominently in the data, perhaps due to the short time period available.

Lastly, we also employed the *STL()* function by statsmodels to obtain a potentially better decomposition. This uses LOESS (Locally Estimated Scatterplot Smoothing) for the decomposition (statsmodels, 2023b).

The decomposition yields a similar result to our removal of the daily trend (ADF test statistic of -35.603 and KPSS test statistic of 0.002). Hence, we stuck with our previous transformation.

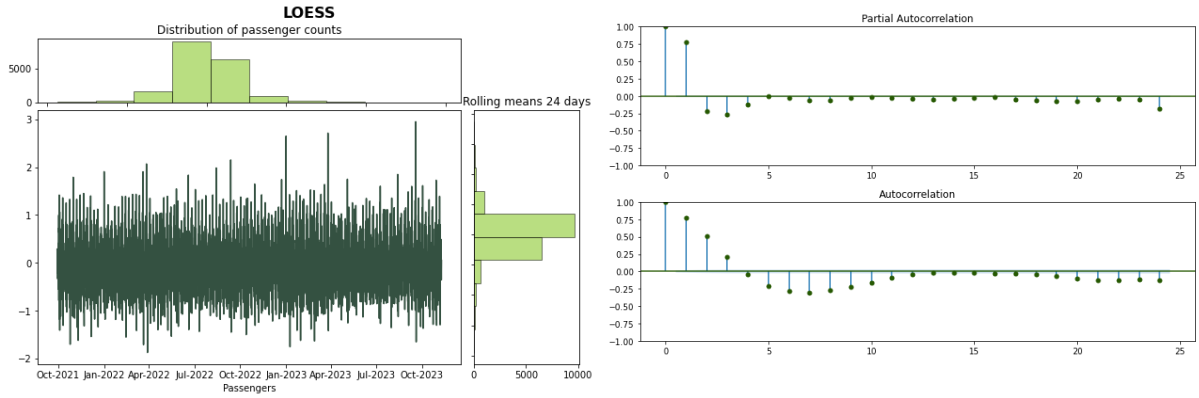


Figure 21: *Visual Analysis of the LOESS De-seasoned Data (Histogram, (P)ACF)*

While we stabilized the mean and the variance of p_t, the ACF still shows a slightly seasonal pattern of approximately 24 lags (the length of a day). This pattern may still be present to a lesser extent after the seasonal pattern was supposedly removed (Chatfield, 2001, p. 32) and suggests that p_t is not a random walk (Peixeiro, 2022, p. 59).

As our own decomposition showed slightly more promising results in the PACF and the ACF compared to LOESS, we stuck with it. Looking at the timeframe of only one week, we see the following result of applying the logarithm and subsequent differencing:

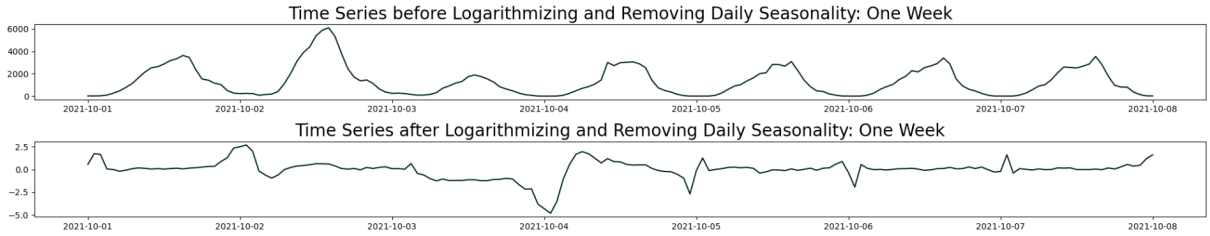


Figure 22: *Before-After Comparison of Taking the Logarithm and De-seasoning*

The daily seasonality that is so prevalent in the upper plot is weakened below. However, when we extend the timeframe to more than a week, we see that there is still a slight weekly seasonality:

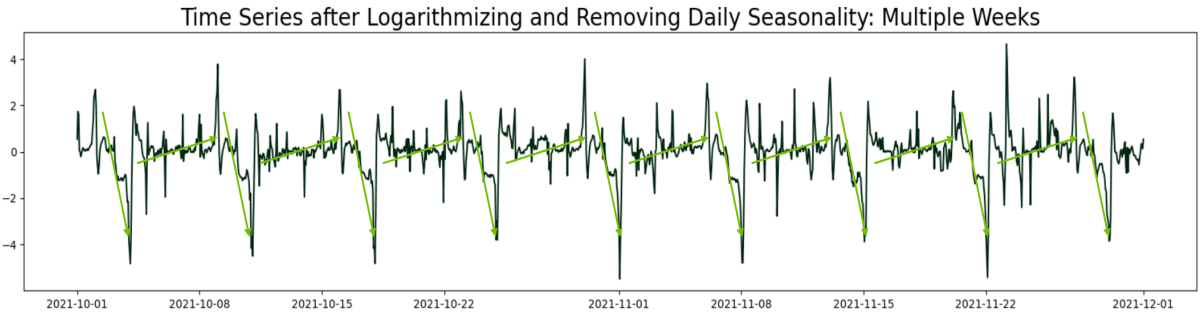


Figure* 23*: *Visualization of Persisting Weekly Seasonality*

The reason for not removing this slight but noticeable weekly seasonality is that – as we realized later on – Lasso would drop almost all of our features if we did, rendering our model useless. Apart from that, the ADF und KPSS test already confirmed the stationarity of our data that was only stripped of its daily seasonality. Therefore, we had to confine with the persisting weekly seasonality.

Inverse Differencing Function

Due to the transformation of our target described above, it is critical to reverse the differencing and logarithmizing after predicting the pedestrian count. We tried out multiple ways to do this, leveraging existing functions like *pmdarima.utils.diff_inv* as well as building our own functions. In the end we inversed the transformation by, first, adding the actual pedestrian count from p_{t-24} to the predicted target $\hat{p}_t$ of our model. Second, we computed $e^{p_{t-24}+\hat{p}_t}$ to reverse the log transformation.

4.3. Status After Preprocessing

Against the background of the chapters above, the following was our starting point for building the model:

Table 2: *Overview of Categorized Variables and the Respective Preprocessing Measures*

Category	Variable	Imputed[5]	Log-transformed	Differenced
Non-binary & hourly	*Pedestrians (Target Variable)*	x	x	x
	Traffic	x	x	x
	Sunshine	x	x	x
Non-binary & non-hourly	*Consumer Price Index*	x	x	
	Retail Trade Turnover	x	x	
	Population	x	x	
	Hotel Guests	x	x	
Binary & hourly	*Weather*	x		x
Binary & non-hourly	*Weekdays*	x		
	School Holidays	x		
	Big Events	x		
	Closed Stores	x		

[5] Imputed only if needed, i.e. if the variable's dataset contained missing values

5. Building the Model

5.1. LSTM Recurrent Neural Network

To predict our target p_t we first chose to build a neural network as it can handle non-linear relationships (Duin, 1996, p. 2). Statistical models, by contrast, may experience performance degradation, especially with daily seasonality and larger datasets (Peixeiro, 2022, p. 176). Additionally, we did not choose a traditional feedforward neural network as they do not sufficiently consider the temporal order of the data and treat features as independent (Kostadinov, 2018, p. 7).

Because p_t is time-dependent, we chose to employ a Recurrent Neural Network (RNN) as it considers past information when processing an element of a sequence (Peixeiro, 2022, p. 213). While traditional RNNs are powerful, we specifically adopted a Long Short-Term Memory (LSTM) neural network to address the issue of vanishing gradients. This can cause higher-order lags to diminish in importance further into the sequence (Peixeiro, 2022, p. 213-214). As our stationary target still shows significant autocorrelation of up to 24 lags, we decided on the LSTM which can retain information for a longer period (Peixeiro, 2022, p. 214).

We first split our features into a training and test set. To avoid the look-ahead bias, we split the data by time (Peixeiro, 2022, p. 29). Then, we standardized the features. This does not only address the challenge of vanishing/exploding gradients but it also facilitates an effective utilization of L1 regularization techniques (Bishop, 1995, p. 296). We have standardized the test data with the mean and standard deviation of the training data. This ensures that the test set is appropriately scaled based on the statistical properties of the training set, maintaining consistency in our model evaluation.

Before we built the model, we applied L1 regularization to determine which features to use as input to the LSTM. We have chosen to apply L1 over Ridge regularization for feature selection as many of our features were not available on an hourly basis. Hence, they may be irrelevant for predicting our target. In contrast to Ridge, L1 regularization can drop these features that do not add value to our model. Therefore, in order to obtain a sparser model, we chose a Lasso regression and determined lambda with cross-validation. Results are shown in *Figure 24*, which illustrates the Lasso coefficients associated with each non-dropped feature.

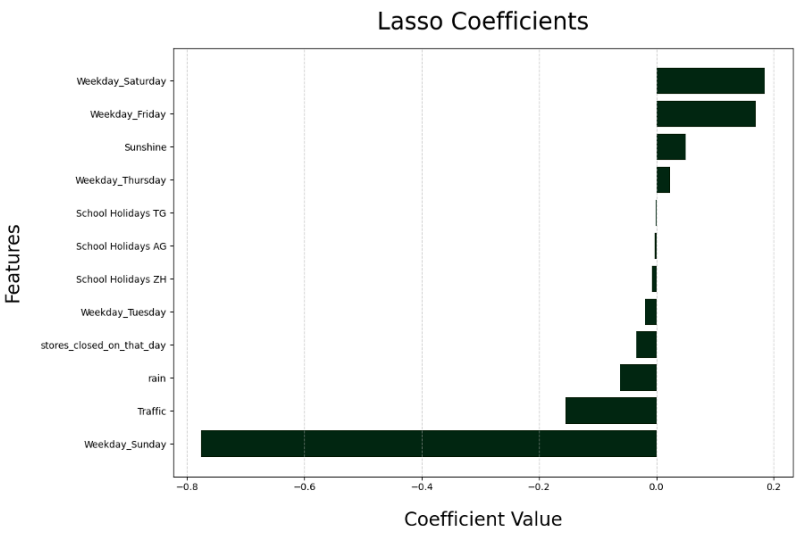


Figure 24: *Coefficients of Lasso Regression of the Target on Our Features*

In total, L1 regularization dropped 24 features, which we have subsequently excluded. Lasso dropped all Public Holiday features and most School Holidays. The most important influence was whether the day was a Friday to Sunday or not. This leaves us with 12 features to train the LSTM on.

Our model is comprised of an input LSTM layer and a series of Dense Layers. To optimize the model's performance, we have selected Adaptive Moment Estimation (Adam) as our optimization algorithm (Kingma & Ba, 2022). To determine the ideal number of neurons, layers, learning rate, and activation functions, we have utilized the *Keras Tuner*, which randomly samples hyperparameters from our defined search space using *Random Search* (Keras, n.d.). Instead of applying L1 or L2 regularization, we added a Dropout Layer to prevent the model from overfitting by learning the statistical noise. The dropout probability was also optimized using *Keras Tuner.* Our loss function is the Mean Squared Error (MSE) to penalize larger prediction errors in the model. We chose to use the MSE and not the MAE as a difference between a dozen of pedestrians may not be relevant for fundraisers. However, a large accumulation of people represents a fruitful opportunity to raise funds while a small number of pedestrians could waste the fundraisers' valuable time. Hence, outliers should be punished more severely. Lastly, to prevent overfitting, we implemented an Early Stopping Monitor that halts the training if there is no improvement in the validation loss for three consecutive epochs (Goodfellow et al., 2016, p. 274).

LSTM With One Time Step as Input

To evaluate how our neural network may perform without considering the temporal dependence of p_t, we first built an LSTM with only one time step as an input. *Figure 25* depicts the corresponding outcomes of the validation and training loss.

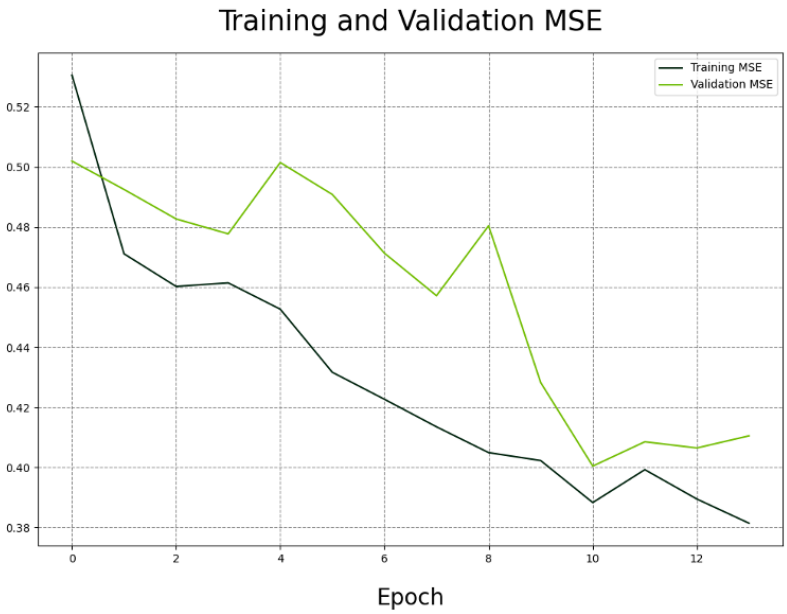


Figure 25: *Visualization of Prediction's Training and Validation MSE*

The simple model already approximates the number of pedestrians and the seasonality of the test data fairly well, as depicted in *Figure 26*. It achieves an MSE of 5'385'744.815. However, due to the temporal dependence of p_t, we went on to build a second LSTM.

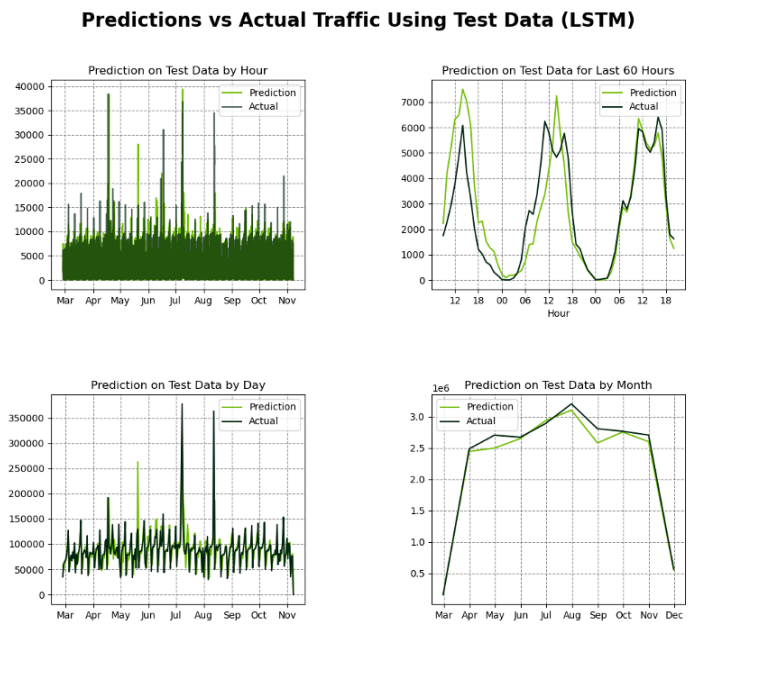


Figure 26: *Comparison of Predictions of the First LSTM and the Actual Traffic*

LSTM With 24 Time Steps as Input

The second LSTM implements a *sequence to vector architecture* (Goodfellow et al. 2016, p. 372). Thus, the network receives multiple observations of the features at once to produce a single output. Since the most important seasonality in our data was observed during the day

and the autocorrelation was not fully removed even after de-seasoning, we have fed 24 time step sequences as input to the model. To enable the model to use information from the sequence, we set *return_sequences* of the LSTM layer to true (Gautam, 2020). As demonstrated in *Figure 27*, the training and validation MSE have significantly improved compared to the previous LSTM.

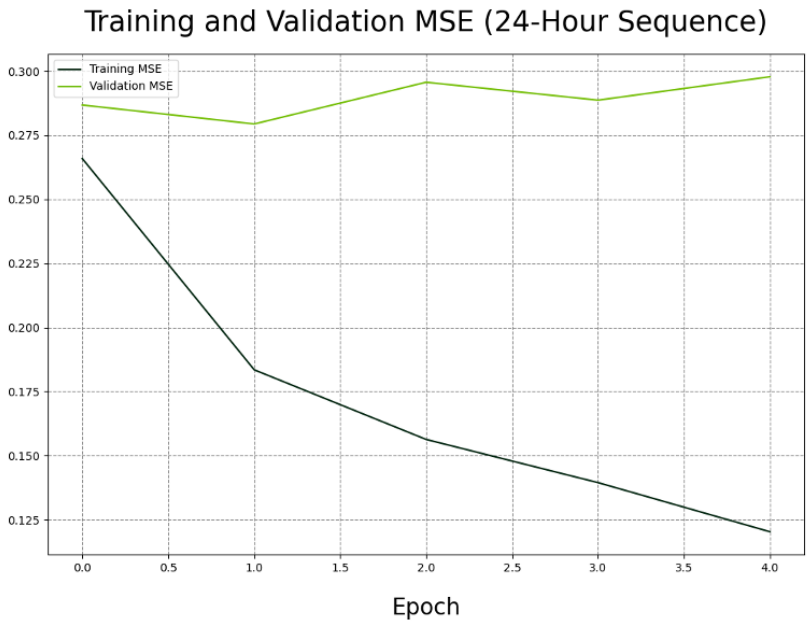


Figure 27: *24h Sequence LSTM Training and Validation MSE*

The predictions on the test data obtained with this model are shown in *Figure 28*. The model achieves an MSE of 3'809'174.394.

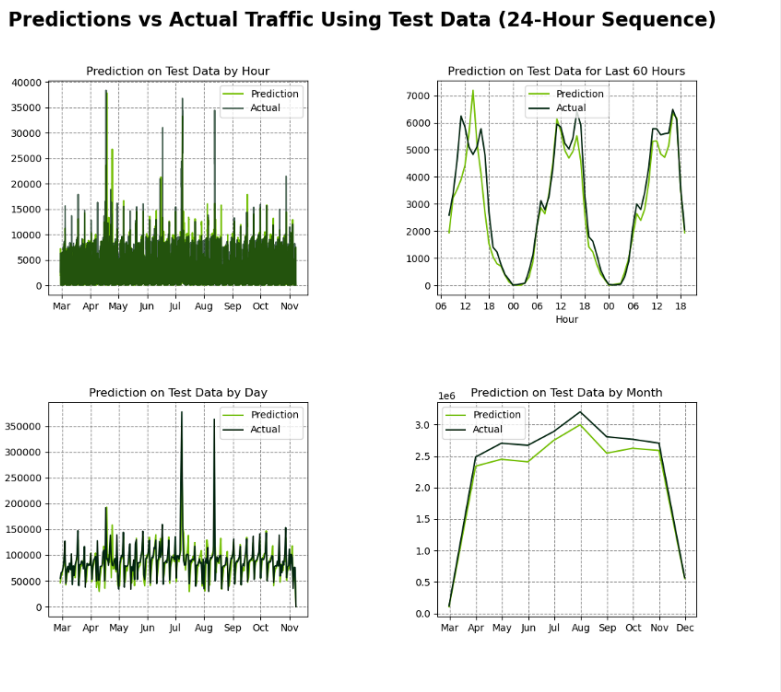


Figure 28: *Comparison of Predictions of the 24h Sequence LSTM and the Actual Traffic*

5.2. Tree-Based Methods

Tree-based models can be used to forecast non-linear time-series data (Rady et al., 2021, p. 229). Hence, we also used tree-based models to predict p_t to compare their performance with that of the LSTM.

First, we modeled a simple Decision Tree to predict p_t. The standardized training and test data is equivalent to that of the LSTM model. As Decision Trees are prone to overfitting, we first applied cost-complexity pruning (Bradford et al., 2005, p. 131). For this, we determined the optimal complexity parameter α using cross-validation. As our target is time-dependent, cross-validation was performed with *TimeSeriesSplit*, which preserves the temporal ordering of the data (scikit-learn, n.d.b). The results of our analysis are shown in *Figure 29*.

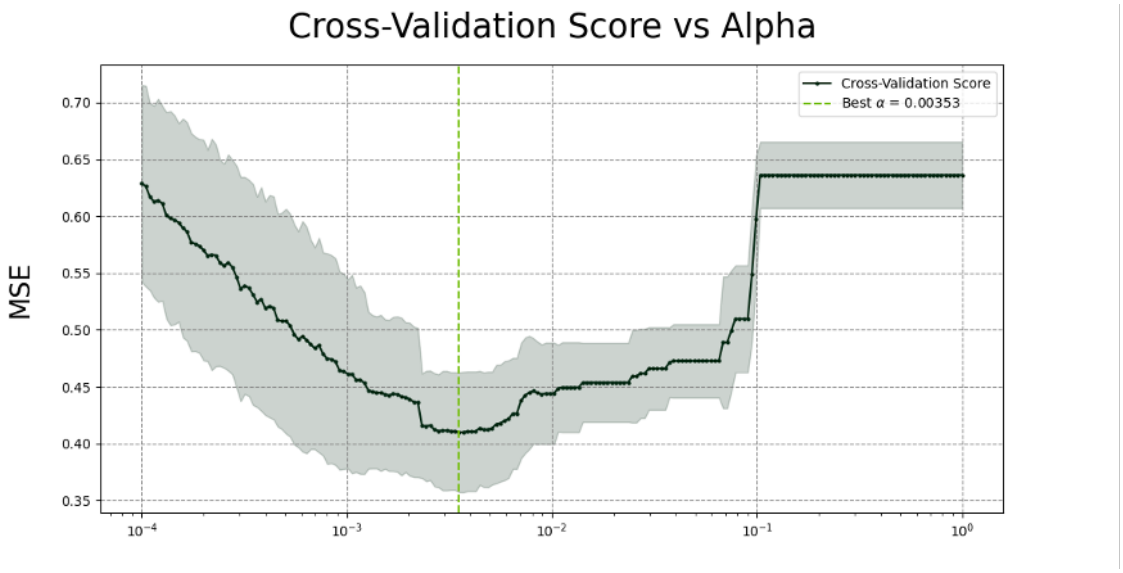


Figure 29: *Optimal Cost-Complexity Parameter After Cross Validation*

Using the α with the lowest cross-validation MSE, we built a decision tree using the *DecisionTreeRegressor*. The first three levels of the tree are shown below in *Figure 30*. They indicate that the traffic feature has an important influence on the predictions of the pedestrian frequency.

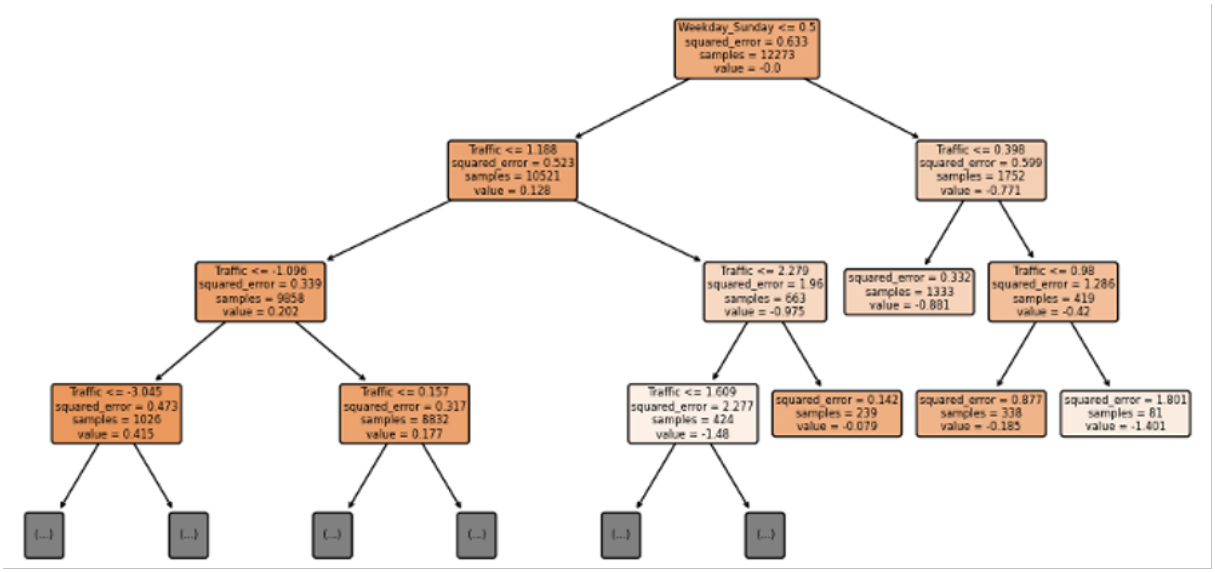


Figure 30: *First Three Levels of the Decision Tree Regressor*

Thereafter, we used the *DecisionTreeRegressor* to make predictions on the test data. When observing the results, we can see that the MSE 5'651'414.36 is higher than the LSTM's 3'809'174.394. To obtain more accurate predictions we expanded on the simple Decision Tree and employed an Ensemble Method, specifically the Random Forest.

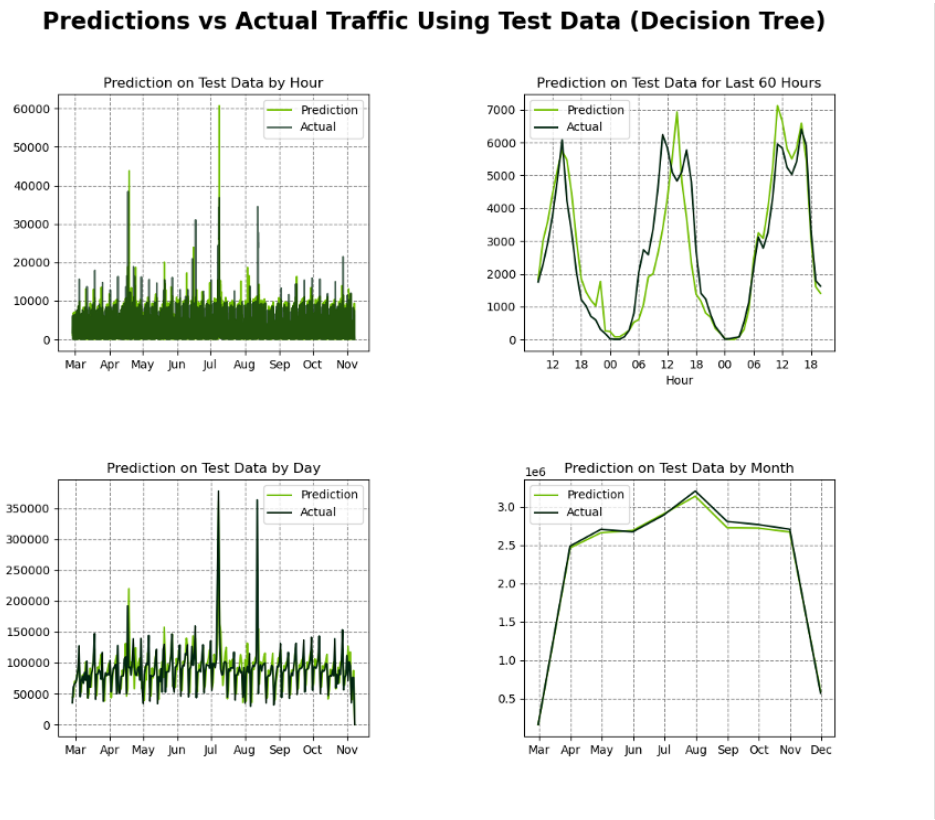


Figure 31: *Predictions on the Test Data Generated by the Decision Tree Regressor*

As the Random Forest trains multiple Decision Trees with variations in selected features and training data, it may reduce the overfitting of the previous *DecisionTreeRegressor* (Rady et al., 2021, p. 240). Training the Random Forest, we first performed hyperparameter tuning by using *GridSearch*. The model with the lowest empirical loss had the following parameters:

i. n_estimators: 200
ii. max_depth: 20
iii. min_samples_split: 10
iv. min_samples_leaf: 4.

By training a Random Forest with these parameters, the following predictions on the test data were obtained:

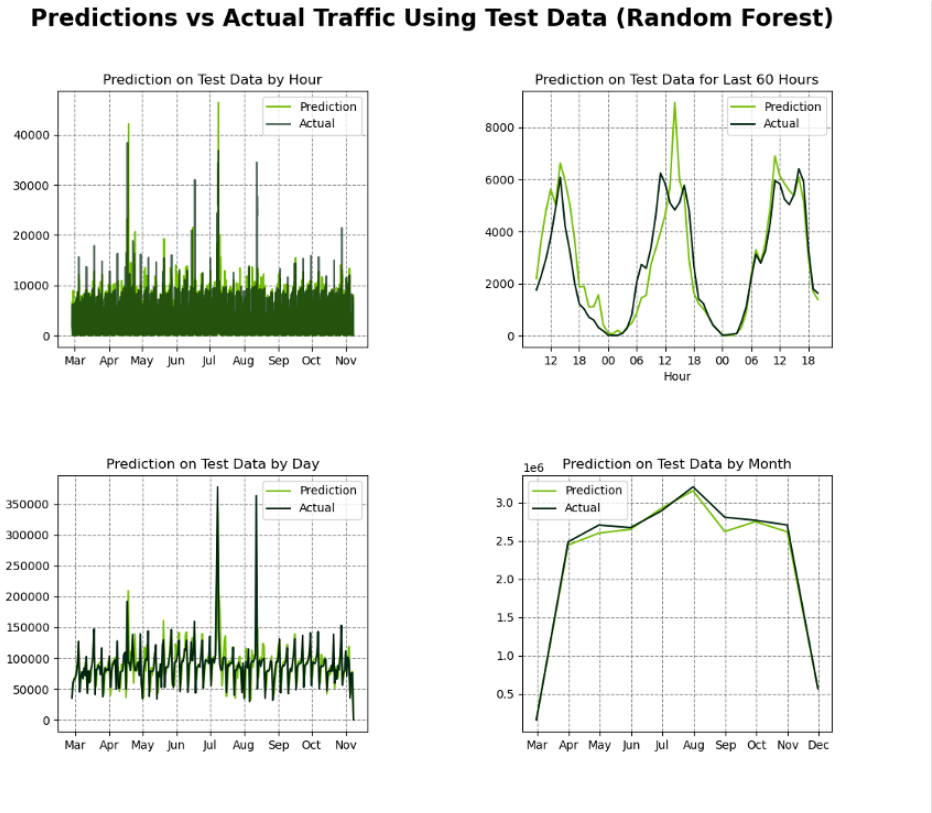


Figure 32: *Predictions on the Test Data Generated by the Random Forest*

While the MSE of 5’034’793.60 is slightly lower than that of the simple Decision Tree, it does again not outperform the LSTM.

6. Conclusion

Throughout the project, we experienced how critical the preprocessing of the data is: Without a well analyzed and appropriately transformed dataset, predictions will turn out worse than otherwise. Moreover, we saw how vital a good coordination is when coding in a team. In order to make our workflow as smooth as possible, we kicked off our project by creating a GitHub repository and agreeing on how to use it. For instance, we discussed how to comment our code. Give that in larger projects, complete coding-chaos is often just around the next corner, this saved us a lot of time. Lastly, we learned of the importance of trying out different solutions, quantifying their results, comparing them and – finally – settling on the best. We leveraged this approach when working on imputation, de-seasoning and building the model. Time and again, we were surprised to see unexpected and rather improvised solutions to achieve the best scores.

Against this background, our project focused to a large part on the preprocessing of the variables, the target variable as well as the numerous features. Scrutinizing them in detail was only the beginning: What followed were individual, sometimes extensive imputations of the missings before the log-transformation was applied and the data was de-seasoned through differencing.

Given the nature of our data as time series, de-seasoning was an element of the project which we were particularly focused on. It was one of our priorities to predict not the overall trend but the very details of the noise in our data. Therefore, we put a great emphasis on finding the best instrument to do so, which ended up being differencing. Subsequently, we went on to work on another aspect of this topic: inverse differencing. After having made predictions, we of course had to re-add the seasonality to our outcome in order to obtain presentable numbers. We worked on different possible solutions to the problem, leveraging existing functions as well as building our own until we settled on a satisfying result.

Next, we standardized the data and applied Lasso which dropped exactly half of our features. At this point, we could start building the model and chose two different, equally promising approaches for our time series: LSTM and Random Forests as an Ensemble Method. Having tweaked them to achieve the best optimized results, we were able to conclude that the 24h-Sequence-LSTM model predicts the most accurately, measured by the MSE.

Looking at our decent predictions, especially with the LSTM approach (see *Figure 28*), our model might be fit to be applied in real-life scenarios. Thus, with the results of our project, fundraisers may now be able to predict reliably when they should go out on the Bahnhofstrasse to raise funds for their cause.

List of References

Air quality measurement, environmental and health protection, Department of Health and the Environment. (2023). *Stündlich aktualisierte Meteodaten, seit 1992.* Retrieved from https://data.stadt-zuerich.ch/dataset/ugz_meteodaten_stundenmittelwerte

Bisgaard, S. (2010). *Time series analysis and forecasting by example.* Hoboken: Wiley.

Bishop, C. M. (1995). *Neural Networks for Pattern Recognition.* Oxford University Press.

Bradford, J. P., Kunz, C., Kohavi, R., Brunk, C., & Brodley, E. C. (2005). Pruning decision trees with misclassification costs. *European Conference on Machine Learning*, 131-136.

Canton of Zurich. (2023). *Gesetzliche, den Sonntagen gleichgestellte Feiertage im Kanton Zürich.* Retrieved from https://www.zh.ch/de/wirtschaft-arbeit/arbeitsbedingungen/arbeitsssicherheit-gesundheitsschutz/arbeits-ruhezeiten/feiertage.html

Chatfield, C. (2001). *Time-Series Forecasting.* Chapman & Hall.

Dickey, D. A., & Fuller, W. A. (1979). Distribution of the estimators for autoregressive time series with a unit root. *Journal of the American Statistical Association*, 427-431.

Duin, R. P. (1996). *Nonlinear data mapping by neural networks.* Retrieved from https://cds.cern.ch/record/400311/files/p11.pdf

Federal Statistical Office. (2023, November 2). *LIK, Totalindex auf allen Indexbasen.* Retrieved from https://www.bfs.admin.ch/bfs/de/home/statistiken/preise/landesindex-konsumentenpreise.assetdetail.29065706.html

Federal Statistical Office. (2023). *Retail Trade Turnover Statistics.* Retrieved from https://www.bfs.admin.ch/bfs/en/home/statistics/industry-services/surveys/dhu.html

Gautam, S. (2020, April 27). *Return State and Return Sequence of LSTM in Keras.* Retrieved from https://sanjivgautamofficial.medium.com/lstm-in-keras-56a59264c0b2

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning (Adaptive Computation and Machine Learning series).* The MIT Press.

Jalil, A., & Rao, N. H. (2019). Chapter 8 - Time Series Analysis (Stationarity, Cointegration, and Causality). *Environmental Kuznets Curve (EKC)*, 85-99.

Keras. (n.d.). *RandomSearch Tuner.* Retrieved from https://keras.io/api/keras_tuner/tuners/random/

Kingma, D. P., & Ba, J. L. (2022, December 22). ADAM: A Method for Stochastic Optimization. *ICLR*.

Kostadinov, S. (2018). *Recurrent Neural Networks with Python Quick Start Guide : Sequential Learning and Language Modeling with TensorFlow.* Packt Publishing, Limited.

Kwiatkowski, D., Phillips, P. C., Schmidt, P., & Shin, Y. (1992). Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? *Journal of Econometrics*, 159-178.

Mushtaq, R. (2011). Augmented Dickey Fuller Test. *SSRN*.

OpenHolidays API. (2023). Retrieved from https://www.openholidaysapi.org/en/#school-holidays

Peixeiro, M. (2022). *Time Series Forecasting in Python.* Manning Publications Co. LLC.

Population Office, Presidential Department. (2023). *Bevölkerung nach Monat, Stadtquartier, Geschlecht, Altersgruppe und Herkunft, seit 1998.* Retrieved from https://data.stadt-zuerich.ch/dataset/bev_monat_bestand_quartier_geschl_ag_herkunft_od3250

Rady, E. A., Fawzy, H., & Fattah, A. M. (2021). Time Series Forecasting Using Tree Based Methods. *Journal of Statistics Applications & Probability*, 229-244.

Safety Department, Traffic Division. (2023). *Daten der Verkehrszählung zum motorisierten Individualverkehr (Stundenwerte), seit 2012.* Retrieved from https://data.stadt-zuerich.ch/dataset/sid_dav_verkehrszaehlung_miv_od2031

scikit-learn. (n.d.a). *Imputing missing values before building an estimator.* Retrieved from https://scikit-learn.org/stable/auto_examples/impute/plot_missing_values.html#sphx-glr-auto-examples-impute-plot-missing-values-py

scikit-learn. (n.d.b). *sklearn.model_selection.TimeSeriesSplit¶.* Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.TimeSeriesSplit.html

statsmodels. (2023a, May 05). *statsmodels.tsa.stattools.adfuller.* Retrieved from https://www.statsmodels.org/dev/generated/statsmodels.tsa.stattools.adfuller.html#statsmodels.tsa.stattools.adfuller

statsmodels. (2023b, May 05). *statsmodels.tsa.seasonal.STL.* Retrieved from https://www.statsmodels.org/stable/generated/statsmodels.tsa.seasonal.STL.html

Urban Deveopment, Presidential Department. (2023). *Passantenfrequenzen an der Bahnhofstrasse - Stundenwerte.* Retrieved from https://data.stadt-zuerich.ch/dataset/hystreet_fussgaengerfrequenzen

Zurich City Presidential Department. (2023, November 3). *Tourismus Stadt Zürich.* Retrieved from https://www.stadt-zuerich.ch/prd/de/index/statistik/themen/wirtschaft/tourismus/tourismus-stadt-zuerich.html

Zurich Tourism. (2023). Retrieved from https://www.zuerich.com/de/besuchen/event-highlights

Declaration of Independence

We hereby declare

- that we have written this term paper without any help from others and without the use of documents or aids other than those stated above;
- that we have mentioned all the sources used and that we have cited them correctly according to established academic citation rules;
- that we have acquired any immaterial rights to materials we may have used, such as images or graphs, or that we have produced such materials ourself;
- that the topic or parts of it are not already the object of any work or examination of another course unless this has been explicitly agreed to with the faculty member in advance and is referred to in the term paper;
- that we will not pass on copies of this work to third parties or publish them without the university's written consent if a direct connection can be established with the University of St.Gallen or its faculty members;
- that we am aware that my work can be electronically checked for plagiarism and that we hereby grant the University of St.Gallen copyright in accordance with the Examination Regulations insofar as this is required for administrative action;
- that we am aware that the university will prosecute any infringement of this declaration of authorship and, in particular, the employment of a ghostwriter, and that any such infringement may result in disciplinary and criminal consequences which may result in my expulsion from the university or my being stripped of my degree.

By uploading this project report, we confirm through our conclusive action that we are submitting the Declaration of Authorship, that we have read and understood it, and that it is true.

No. of Characters: 42'428

10.12.2023