Business Analytics: Data Retrieval and Analysis in R and Python

– Seminar Winter Term 2018/19 –

Analysis of crime in Chicago: What's behind Chicago's surge in violence?

Submitted by:
Hadi Chami

Student-ID:
3341371

Advisor:
Prof. Dr. Dirk Neumann
Dr. Gunther Gust

Contents

Abstract

In no other major US city do so many people die a violent death as in Chicago. Will the number of crimes increase in the future? To answer this and further important question this paper explores a broad insight the progress of criminality in the city of Chicago. The given dataset from Kaggle was used to develop a shiny dashboard that provides an overview on the content of the data based on interactive plots. Further, a prediction of the criminality trend has been analyzed with the prophet package by Facebook. The visualization and statistical analysis give clear indications that the city is on a superior way. Therefore, we can hold on - it can be recorded that the crime rate in the city of Chicago will be decline within the next few years.

1 Introduction

For approximately a century sociologists have been trying to comprehend the spatial distribution of crime in urban neighbourhoods. Chicago has been actually notorious for being the crime capital of the United States since the days of organized crime at the same time as the prohibition were introduced. Moreover, the rapid urbanization and the socio-economic heterogeneity of the city makes it an optimal case for the examination of relations among urbanism and crime. The Chicago School of Sociology is acutely advanced relating to developments and teachings on the area and has been widely inspired by analysis performed in the fields of economics and sociology (4).

From the beginning of the 90s, the City of Chicago has been carried out a range of crime prevention policies. These arrangements seem to have been effective since Chicago dropped out of the main 25 most unsafe cities in the U.S. in 2006. However, the number of murders still persist higher than in other cities as New York City or Los Angeles and the current rise in crimes occurred in Chicago brings the present anti crime policies into question. Furthermore, the public policies are confronted with the economic situation, because where there are high levels of crime, there also will be a high level of structural weakness - like: poverty, segregation and residential instability (6).

In this context, the aim of this paper is to determine the trend of the violent crime in the city of Chicago over the 2001-2016 period. More specifically, several analysis provide in-depth information on different crime issues in Chicago. Therefore, a brief motivation and the originated research questions will be explained in section 2. Section 3 examines the dataset that was used for the descriptive analysis and provides information about the data preparation and cleaning process. Section 4 describes the prophet package and the theoretical and mathematical composition of the the model. The practical application and analysis of the forecast will be presented in Section 5. In Section 6 the results from of the descriptive analysis and the prediction will be explained and relations examined. Following, a conclusion will be given.

2 Research Question

In order to analyze the data of the given dataset it is of great interest to establish several research questions to provide deep insights into the evolution of crimes in the city of Chicago. Broad research concerning the crime trend in Chicago and the associated crime rate already exists and exhibit deep insights into the negative development of the city. Andrew Papachristos describes in his paper that Chicago is known as the murder

capital of the U.S. within a violent rate of 1045.15 per 100,000 resident. Nevertheless, based on the data from the United States Department of Justice and Chicago Police, the overall rate of crime and violent has significant drop down over the last four and a half decades (3). To follow the findings of Papachristos, a very interesting aspect would be to see how the trend of the crime rate will be evolve in the future, done through a forecast of the given data.
However, in addition to the declining crime rate, an increase in homicide has been prognosticated by Griffiths and Chavez. The reason for the sudden increase is related to the drug market expansion, demographic changes and rise in handgun usage (5). Based on the finding of Griffiths and Chavez, it is interesting to see what kind of crime types occur most in the city of Chicago, in order to have an overview about Chicago's criminals intension. Moreover, it would be also important to know where most crimes happen, which let us know where safe places are.
Harry Willbach made a breadth analysis and research about arresting in Chicago, and compared different types of crime with the associated arrest rate. Willbach figured out, that the number of persons arrested for the several groups of crimes not following any pattern, but indicates a change - in creasing and decreasing (16). Therefore, we will analyse how successful the police in Chicago has been in arresting criminals and how they have change over time. We take the described ideas to develop the following research questions:

1. How is the current crime rate in Chicago and how it will develop in the future?
2. What are the most common crimes in Chicago from 2001-2016?
3. What are the locations were a high number of recorded crime are reported?
4. With the reported crimes, were arrests made or not and how does the trend look like?

3 Empirical Design

The provided dataset was extracted from the Chicago Police Department's CLEAR (Citizen Law Enforcement Analysis and Reporting) system and deliberate reported incidents that occurred in the City of Chicago. For the privacy protection of the crime victims, addresses are just appear at the block level only and particular locations are not identified. The dataset was obtained from Kaggle and covers a timeline from 2001 to 2016. Kaggle is an online community of data scientists and machine learners owned by Google Inc.

3.1 Dataset

1. Dataset Preparation

The first step was to load the data into R, which can be done by using the *read.csv()* function.

```
c1 = read.csv("Chicago_Crimes_2001_to_2004.csv", header = TRUE)
c2 = read.csv("Chicago_Crimes_2005_to_2007.csv", header = TRUE)
c3 = read.csv("Chicago_Crimes_2008_to_2011.csv", header = TRUE)
c4 = read.csv("Chicago_Crimes_2012_to_2017.csv", header = TRUE)
```

After all four files have been successfully loaded, they were merged together into one big dataset (cc) through the *rbind()* function.

```
cc <- rbind(c1, c2, c3, c4)
```

2. Dataset Cleaning

The second step was to tidy the data. It is one of the important cleaning processes during big data processing and is a recognized step in the practice of data science. All unnecessary columns for the data analysis are taken out by using the *select()* function of the dplyr package.

```
select(cc,-c(ID,Case.Number,IUCR,Domestic,Beat,District,Ward,Community.Area,
             FBI.Code,X.Coordinate,Y.Coordinate,Updated.On,Location))
```

Only the most important data was kept, which now will be explained in detail in order to understand the descriptive analysis procedure.

Date: Date when the incident occurred. Because we are dealing with dates, we needed to convert the Date column.

Block: The partially redacted address where the incident occurred, placing it on the same block as the actual address.

Primary.Type: The primary description of the IUCR code, which illustrates different crime types.

Location.Desc.: Description of the location where the incident occurred.

Arrest: Indicates whether an arrest was made or not.

Latitude:	The latitude of the location where the incident occurred.
Longitude:	The longitude of the location where the incident occurred.

3.2 Descriptive Analysis

1. Practical Application

In this section, the practical application with the help of shiny in R will be presented. In order to visualize and illustrate different analysis of the dataset, a shiny dashboard has been developed. First a logical structure of the program is given, that briefly describes the main idea behind the construction of the app. Second, the hole output of each page of the shiny dashboard, represented in the form of code snippets and plot output, will be explained and illustrated.

2. Program Structure

Shiny is an open source R package that allows it easy to visualize interactive web-based applications straight from R. The shiny package provides a powerful web framework and comes with a variety of widgets for quickly building user-interface without requiring any HTML, CSS or JavaScript knowledge (7). For the seminar paper, the shiny dashboard app is structured into four main menu items and two subitems. Each menu item represents a different visualization of the dataset and shows the use of various powerful and creative functions of the shiny package.

The application is divided into the following four pages:

1. Time Series
    - Crimes and Arrests
    - Forecast
2. Visualization
    - Crime Types by Year
    - Crime Locations by Year
3. Map
4. Heatmap

The following section explains the framework and technical implementation of the shiny dashboard application. Each individual menu item is described and explained by detail with code snippets and graphical outputs (B). Furthermore, it is possible to access the shiny dashboard through the provided link next to each explained menu point in order to simultaneously apply the theory in practice.

3. Program Output

To navigate through the shiny application the following link can be used:

`https://hadichami.shinyapps.io/Crimes_Chicago_Shiny/`

1. Time Series (`https://hadichami.shinyapps.io/Crimes_Chicago_Shiny/`)

To see how crimes are spreaded out over the time, a time series analysis was realized that illustrated the amount of crimes and arrests over the years. Furthermore in order to predict how the crime rate in Chicago will shift in the future, a forecast was made within the prophet package that is explained in Section 4.

1.1 Crimes and Arrests

The Crimes and Arrests submenu shows crimes and arrests from 2012 to 2016 based on two time series. The first step was to filter the data for a time period of 2012 till 2016 and then to create a time series by using the *xts()* function of the xts package (8). To plot the time series in shiny the *hchart()* function was used, where an arrest time series was also added to.

Figure 6 illustrates all crimes from 2012 till 2016 and all arrests for this period. To detect how arrests changed over the years, Figure 7 provides insights. Both output and the used code snippet can be seen in the Appendix Section B.

2. Visualization (`https://hadichami.shinyapps.io/Crimes_Chicago_Shiny/`)

The Visualization page is used to represent different visualizations of the dataset and to explain and answer, based on interactive plots, the provided research questions from Section 2. For the graphical representation not the usual ggplot package was used, rather the highcharter package. Highcharter, allows a rich R interface to the popular Highcharts JavaScript graphics library and besides the hover effect also comes with an integrated download function to export the output in various formats.

2.1 Crime Types by Year

The submenu Crime Types by Year demonstrates how high the crime is and what kind of different crime types exists. Based on two filter widgets it is possible to manipulate the data, either by selecting a specific crime type or by picking a time period. To create and plot in shiny the *hchart()* function was also used here. Within the function different arguments can be used to determine the representation, color, title, export and much more.

Figure 8 shows the crime rate of the type Theft from 2001 to 2016. To see what types of crime occurred most in the city of Chicago, Figure 9 provides an overview (see Appendix B).

2.1 Crime Locations by Year

The second submenu Crime Locations by Year indicates how high the crime of the different crime locations are. Again, the same filter widgets and plot output function were used as in the section before.

Figure 10 illustrates all crimes happened on the street from 2001 to 2016. To see in which locations most crime occurred, Figure 11 provides an overview (see Appendix B).

3. Map (`https://hadichami.shinyapps.io/Crimes_Chicago_Shiny/`)

The following part of the app exhibit, based on a map of Chicago, on which place crimes occur in the city. For the use of an interactive map the leaflet package is a powerful tool and comes with a lot of extra features that can be integrated and used. The basic usage of the leaflet map is structured into three parts:

1. Create a map widget by calling leaflet()
2. Add layers to the map (e.g. addTitles(), addMarkers())
3. Print the map

Markers that are spread all over the map indicate the amount of crimes in this area. By clicking on a marker the map will automatically zoom in and spread the markers on a detailed level. This step is complete until a single point (crime) is displayed on the map where a pop-up element shows the detailed crime information.

Figure 12 shows the map of Chicago with circle markers that represents the amount of crimes in this area for a given crime type. By zooming inside the marker, a single crime is displayed within a pop-up window that includes the crime information (Appendix B).

4. Heatmap (`https://hadichami.shinyapps.io/Crimes_Chicago_Shiny/`)

In addition to the previously described map, a heatmap is used to give an intuitively and quickly overview into a large amount of data and to make particularly striking values easily recognizable. This heatmap works with different colors, which are typically assigned to temperatures. The classic heat map consists of the colors red (for very hot), orange (for medium hot), yellow (for warm), green (for cool), blue (for cold).

Figure 13 illustrates a heatmap of Chicago where based on temperature the most crimes are colored in red (see Appendix B).

4 Prophet Forecasting Model

For data scientist forecasting is an essential instrument to predict future movements based on past and present data. Organizations across all sectors of industry have to employ in capacity planning to use scarce resources and goals efficiently and effectively. Although it is an important tool - there are often challenges in creating and choosing the appropriate forecast model (13).

Prophet is an open source R package that allows it easy to forecast time series data where non-linear trends are fit with yearly, weekly, and daily seasonality, and holiday effects. It was created and released by Facebook's Core Data Science team and is available on CRAN and PyPI (14). Prophet performs best, when strong seasonal effects are present and a few periods of recorded information are existent. Furthermore, prophet can deal with missing data effectively and treats outlier very well.

The underlying model behind the forecast is a decomposable time series model within three main model components: trend, seasonality and holidays.

The following equation describes the model:

$$y(t) = g(t) + s(t) + h(t) + \epsilon t \tag{4.0.1}$$

$g(t)$ represents the trend function, $s(t)$ describes periodic changes (e.g. weekly or yearly seasonality) and $h(t)$ typify the holiday effect that arise on irregular schedules. The error term ϵt represents all peculiar changes that are not considered by the model (13). The just described model is equal to a generalized additive model (GAM) where the linear predictor depends linearly on unknown smooth functions of some predictor variables (15). Prophet is constructed in such a way that it exclusives uses the time as a

regressor, however, several linear and non-linear functions can be extended. Modeling seasonality as an additional element is the identical approximation taken by exponential smoothing (9).

In the following subsections the components of the prophet model are described in detail in order to understand the mathematical background behind the model. First, the trend model, especially the saturating growth and linear model are presented. Second, the seasonality component is explained to understand how to deal with shifts happened in the series. Third, the holidays and events, that provide large predictable shocks to time series are clarified.

4.1 Trend Model

A trend is a systematic increase or decrease in a time series that also exhibits fluctuation or periodic fluctuations. Prophet consists two models behind their trend component $g(t)$: a non-linear saturating growth model and a linear model.

1. Saturating Growth Model

In order to analyse growth forecasting, the saturating growth model is used to detect the growing procedure in the past and future. It is modeled by using the logistic growth model, which in its standard form is:

$$g(t) = \frac{C}{1 + exp(-k(t-m))'} \tag{4.1.1}$$

where C represents the carrying capacity, k describes the growth rate and m is an offset parameter. Beside the basic logistic growth model, there are two essential aspects of growth that are not determined in Equation 4.1.1. The carrying capacity is not an constant factor and increases with the number of observation. Thus, the capacity component gets an additional time-varying variable $C(t)$ that replaces the fixed capacity. Furthermore, the growth rate is likewise not constant. New products can essentially change the rate of growth, so the model should be able to take over an alterable rate instead of a constant rate.

Trend changes in the growth model are realized by specifying changepoints where the growth rate is able to move. Assume a number of S changepoints at time $s_j, j = 1, .., S$. Further, a vector of rate of adjustments $\boldsymbol{\delta} \in \mathbb{R}^S$ is created, where δ_j is the adjustment in rate that happens at time s_j. The rate at time t is made up of the base rate k and all

adjustments toward this point: $k+ \sum_{j:t>sj} \delta_j$. This is exhibited in detail by clarifying a vector $a(t) \in \{0,1\}^S$ such that

$$a_j(t) = \begin{cases} 1, & \text{if } t \geq s_j, \\ 0, & \text{otherwise.} \end{cases}$$

The rate at time t is described as $k+a(t)^T\delta$. Every time the rate k is adapted, the offset parameter m must also be modified to link the endpoints of the segments. The correct adaption at changepoint j is measured as

$$\gamma_j = \left(s_j - m - \sum_{l<j} \gamma_l\right)\left(1 - \frac{k + \sum_{l<j} \delta_l}{k + \sum_{l\leq j} \delta_l}\right)$$

The logistic growth model is then

$$g(t) = \frac{C(t)}{1 + exp(-(k + a(t)_T\delta)(t - (m + a(t)_T\gamma)))} \tag{4.1.2}$$

The introduced logistic trend model in 4.1.2 is a specific form of a generalized logistic growth curve, which is just an individual kind of sigmoid curve. Expansions of this trend model to different groups of curves is under rapidly work (13).

2. Linear Model

In order to analyse forecast problems without an existing progress or growth in time, the linear trend model with a constant rate of growth is used instead of the mentioned saturating model. Here the trend model is given as

$$g(t) = (k + a(t)_T\delta)t + (m + a(t)_T\gamma) \tag{4.1.3}$$

where k represents the growth rate, δ stands for the rate adjustments, m is the offset parameter, and γ_j is determined to $-s_j\delta_j$ to make the function proceed (13).

4.2 Seasonality

In time series, seasonality are the fluctuations that appear at frequent intervals, such as weekly, monthly or yearly. Reasons for the occurrence of seasonality can be several, for instance the weather, vacation, holidays and periodic regular patterns (11).

In order to provide a flexible model, fourier series were used to facilitate periodic effects (2). We assume that P is the regular period that the time series may take (e.g. $P = 365.25$ for yearly data or $P = 7$ for weekly data). Arbitrary smooth seasonal effects are modeled

$$s(t) = \sum_{n=1}^{N} \left(a_n cos\left(\frac{2\pi nt}{P}\right) + b_n sin\left(\frac{2\pi nt}{P}\right)\right)$$

with a standard Fourier series. To fit seasonality the $2N$ parameters $\beta = [a_1, b_1, ..., a_n, b_n]^T$ needs to be estimated by establishing a matrix of seasonality vectors for every value of the historical and future data. For a yearly seasonality and $N = 10$ we observe the following equation

$$X(t) = \left[cos\left(\frac{2\pi(1)t}{365.25}\right), ..., sin\left(\frac{2\pi(10)t}{365.25}\right)\right] \tag{4.2.1}$$

where the seasonal component is then

$$s(t) = X(t)\beta \tag{4.2.2}$$

represented as a standard Fourier series. For the generative model the use of the β ~ $Normal(0, \sigma^2)$ was applied in order to obtain a grading on the seasonality. Cutting of the time series at the point N leads to a low-pass filter to the seasonality. Thus, every time N increases, seasonal patterns are fitting rapidly, however the risk of overfitting increases too. The seasonal component in particular yearly and weekly seasonality works best with $N = 10$ and $N = 3$ (13).

4.3 Holidays and Events

The third component behind the prophet model are holidays or events that occur during the year and provide predictable shocks which do not correspond to a pattern. The holiday shock appear usually year after year, so it is relevant to integrate it into the forecast. Integrating the holidays into the model are accomplished by assuming that impact of holidays are self-contained. Thus, each holiday i, is part of D_i - a set of past and future dates for that single holiday. Further, an indicator function verifies whether time t happens on a holiday i, and if so, assign every holiday to a parameter k_i that

is the associated change in the forecast. This is handled similarly to the seasonality component by developing a matrix of regressors

$$Z(t) = [1(t \in D_1), ..., 1(t \in D_L)]$$

and taking

$$h(t) = Z(t)k \tag{4.3.1}$$

Neither the holiday component uses a prior k ~ $Normal(0, \nu^2)$. A major part is that the model includes a range of days around a specific holiday. To take that into account, further parameters for the days around the holiday are implemented, in order to handle these days as a holiday too (13).

4.4 Model Fitting

For modeling a combination of the seasonality and holiday component for each observation are represented in a matrix X and the changepoint indicators $a(t)$ in a matrix A. The model fitting procedure is based on Stans L-BFGS in order to detect a maximum subsequently estimate, but can also perform a full posterior inference to incorporate the inaccuracy of the model parameters into the forecast uncertainty.

Stan code for the complete model

```
model {
   // Priors
   k   normal(0, 5);
   m   normal(0, 5);
   epsilon   normal(0, 0.5);
   delta   double_exponential(0, tau); beta   normal(0, sigma);
   // Logistic likelihood
   y   normal(C ./ (1 + exp(-(k + A * delta) .* (t - (m + A * gamma)))) + X * beta, epsilon);
   // Linear likelihood
   y   normal((k + A * delta) .* t + (m + A * gamma) + X * beta, sigma);}
```

An important advantage of the decomposable model is the ability to inspect each component of the forecast independently. Through this advantage, analysts are able to get a broad look into their forecasting problem, than simply delivering a forecast. In order to modify the changepoints and seasonality of the model, the parameters tau and sigma are controls for the amount of regularization, which is an important task to avoid overfitting of the model (13).

5 Practice with Prophet

In this section the application of the model illustrated in section 4 will be discussed. For the forecast we build a model based on the variables available in the dataset. Thus, in order to see how the trend of crimes in Chicago will be in he future, the equation takes place as follows:

$$CrimeRate = Trend + Seasonality + Holiday + ErrorTerm$$

where crime rate is represented by y of the df dataset and the trend, seasonality, holiday and error term illustrated by ds.

5.1 Forecast Modeling

The forecast section is separated into one filter widget, the forecast output and a download section where the current forecast can be download in various formats. The main idea was to interactive change the plot based on the selected prediction year.

```
sliderInput("per", label = "Select Period", min = 1, max = 6, step = 1, value = 3)
```

In order to use prophet for the prediction, first a data frame (”df”) with two columns has been created. Further, the column names were changed into ”ds” and ”y” and converted into a factor data type.

```
df <- Year12_16 %>% group_by(Date2) %>% summarise(y = n()) %>% mutate(y = log(y))
names(df) <- c("ds", "y")
df$ds <- factor(df$ds)
```

Next, the *prophet()* function (10) is used to to fit the model. Prophet is doing that in just one line of code.

```
m <- prophet(df)
```

After the model is fitted, the *make_future_dataframe()* function creates a dataframe with future dates for forecasting. The periods arguments indicates the number of future forecasts in years, which is represented here as 365 (one year)*manual input in the shiny app (between 1 and 6).

```
future <- make_future_dataframe(m, 365*input$per)
```

The *predict()* function is used to forecast the given historical datasets.

```
forecast <- predict(m, future)
tail(forecast[c('ds', 'yhat', 'yhat_lower', 'yhat_upper')])
```

To plot the model the *dyplot.prophet()* function is used which generates an interactive plot to hover over the forecast and read the values.

```
dyplot.prophet(m, forecast)
prophet_plot_components(m, forecast)
```

Prophet also provides insights into the components of the model. Thus, the overall trend, weekly and yearly seasonality can be visualized (13).

```
prophet_plot_components(m, forecast)
```

The result of the forecast through the prophet package is illustrated in Figure 1. The overall trend, weekly and yearly seasonality can be shown in Figure 2.

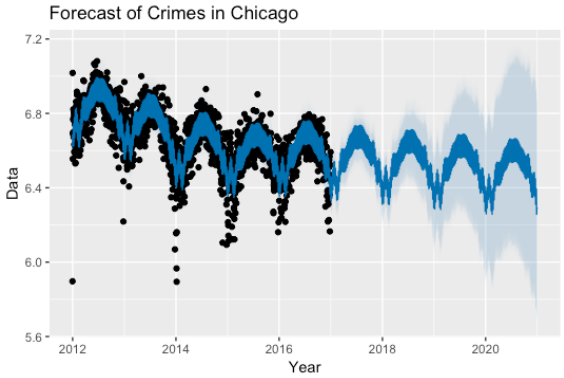

Figure 1: Forecast of Crimes in Chicago.

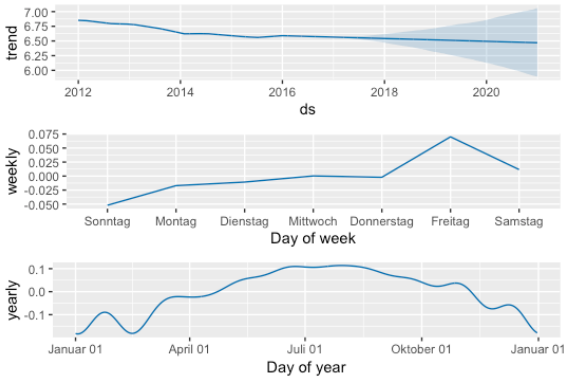

Figure 2: Components of the model.

5.2 Hold-out Sample

Out-of-sample testing is a popular way to test the likely accuracy of a forecasting method. For this purpose, the hole data has to be split into a train and test set. First, the model is fitted and trained on the training data. Second, the trained model is tested and evaluated through the test dataset. The split is necessary to avoid overfitting of the model, which would occur by using the hole data. In our case, a 70/30 split was performed (12).

```
df <- Year12_16 %>% group_by(Date2) %>% summarise(y = n()) %>% mutate(y = log(y))
names(df) <- c("ds", "y")
df$ds <- factor(df$ds)
train_size <- nrow(df) *0.7
trainData <- df[1:train_size,]
testData <- df[(train_size+1):nrow(df),]
```

Thus, after splitting the data the model can be trained and predicted with the actual value and forecast data.

```
m <- prophet(trainData)
p <- predict(m, testData)
plot(m, p)
```

Figure 3 represents the output of the trained model and illustrates the idea behind the hold-out sample.

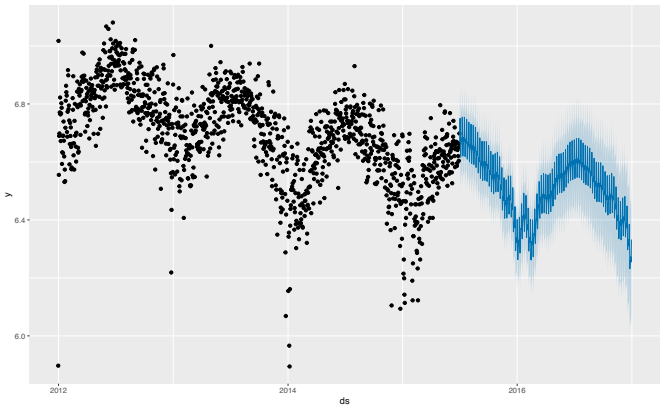

Figure 3: Trained model with test data forecast.

5.3 Diagnostics

Prophet comes with various functions to analyse future predictions error. In order to measure the accuracy of a forecast the cross validation function can be used. Here, the time series is divided into an inital, cutoff and horizon part, that can be defined as parameters within the function. The procedure starts by selecting cutoff points, and fitting the model for each of the points only up to that cutoff. Afterwards the prediction can be compared to the actual values. The split into the mentioned three parts can be seen in Figure 4.

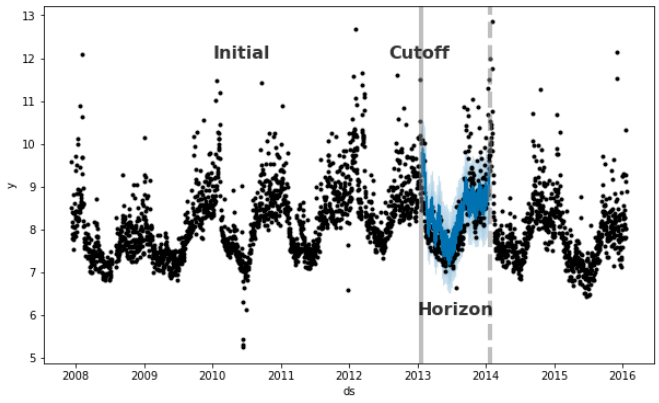

Figure 4: Cross validation split.

The *cross_validation()* function is used on a horizon of 180 days, starting with an inital value of 540 and making predictions every 90 days. For this time period, this leads to 7 total forecasts.

```
df.cv <- cross_validation(m, initial = 540, period = 90, horizon = 180, units = 'days')
```

Further, to determine the forecast error, the *performance_metrics()* function can be used. Here, useful statistics of the prediction performance are illustrated and computed the mean squared error (MSE), root mean squared error (RMSE), mean absolute error (MAE), mean absolute percent error (MAPE) (1).

```
df.p <- performance_metrics(df.cv)
```

The following table illustrates the above mentioned statistics, that represents the mean of the particular error.

horizon	mse	rmse	mae	mape
180 days	0.01971	0.13706	0.10427	0.01591
%	1.97%	13.70%	10.42%	0.015%

Table 1: Performance metrics with the respective forecast errors.

To visualize the cross validation performance metrics the *plot_cross_validation_metric()* function shows the absolute percent error for each prediction in df_cv (1).

```
plot_cross_validation_metric(df.cv, metric = "insert forecast error short name")
```

Figure 5 illustrates the above calculated forecast errors mse, rmse, mae and mape.

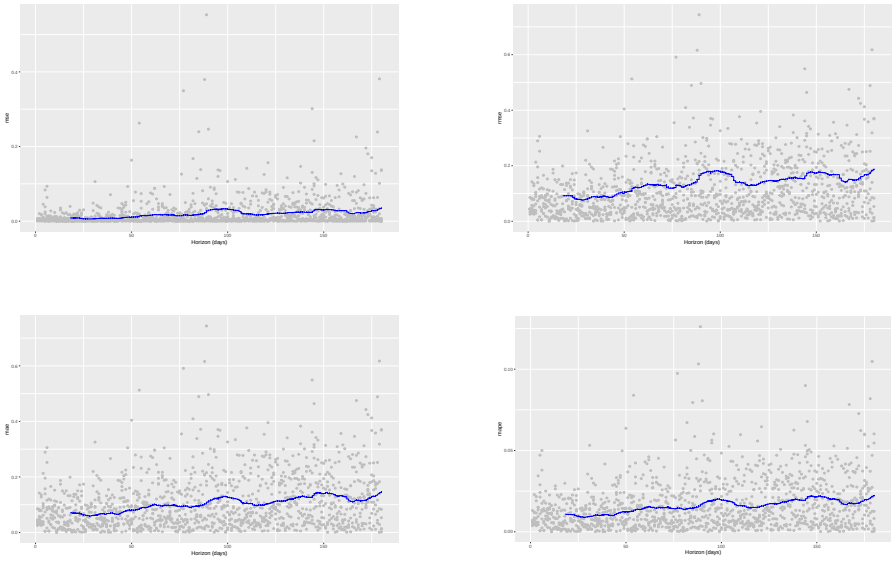

Figure 5: Cross validation performance metrics.

6 Results

In this section the results of the seminar paper will be presented and discussed in detail. For this purpose, the research questions from the motivation section are answered on the basis of analysis and graphical visualizations.

From the time series plot of Figure 6 it is obvious that more crimes were committed than arrests were made. There is clear indication in the time series that crime numbers increase during middle of the year mostly during summer months and drops down at the end of the year especially around the winter months. In total consideration there is a clear decreasing trend of the crime rate. Further, Figure 7 shows a decreasing number of arrests in total from 2012 to 2016. However, by looking at the time series spread over a year, it can be seen that during the spring until summer the arrest rate rises and then falls down again during the winter time.

Figure 9, is sorted descending from the highest crime rate of the top 20 crime types. There were more theft crime incidents about around 12.8 percent compared to battery crimes. Furthermore, theft crimes were extremely high between 2006 and 2010, but have dropped dramatically since 2011 by almost 50 percent. Not only theft crimes decreased, also all other crime types as well. That clearly shows that the police of Chicago succeeded in preventing crimes in the city beginning from 2011.

Another analysis refers to the top 20 locations where most crimes occurred, which is illustrated in Figure 11. Here, most crime happened on the street. Compared to Residence which is the second highest crime location under the top 20, crimes on the street are 40 percent higher. Furthermore, even an around 88 percent higher crime rate compared to crimes happened in apartments.

The map of Chicago illustrated in Figure 12, gives an overview at which location in the city of Chicago crimes occurred. Based on the provided GPS data of the crime incident, it can be examined which places are considered safe or unsafe. For a faster detection which places in Chicago are safe or where the crime rate is high, a second map was additionally added that clearly works with heat temperature colors. Thefts are mostly happening on the street and are compared to locations as residence or apartment, very high.

A very important part of the seminar paper is the future prediction of crimes in Chicago. As we already described the theoretical background and the model in Section **??**, we now interpret the forecast and result. The forecast provided by the prophet package, shown in Figure 1, clearly shows that a decreasing trend will take place in the future

and so the crime rate will be fall. Furthermore, it is evident that most crimes take place in the middle of the year, particularly in the summer months. It could be traced back to the fact that the weather is really good and many people spend a lot of time outside. Another very powerful feature of the prophet package is illustrated in Figure 2, where the components of the model are represented. Thus, again the trend for the coming years is shown a decreasing shift. Further, the second graph shows that most crimes will probably occur at the end of the week, specially on Friday. Finally, the third component illustrates the distribution of how crimes will take place over the year. It is obvious that as already mentioned in the middle of the year most crimes will take place.

In order to measure how well the model performs, the different forecast errors from Table 1 can be used to evaluate the performance. In addition to the four recorded error, the root mean square error (rmse) is particularly used to rate the model. With the selected parameters from the accuracy forecast function, we get a rmse of 0.13706 (13.70%). Here, the difference between the predicted values and the actual values is measured. Thus, the model performance very well with an rmse of 13.70%. and the hold-out sample exhibits a good fit of the trained model. Figure 5 illustrates the graphical representation of the mse, rmse, mae and mape. It is clear that the values confirm a stable model and thus a good performance can be accredited.

7 Conclusion

The provided dataset of the city of Chicago gives a broad insight of the criminal behaviour from 2001 till 2016. The mentioned negative criminal trend in the Introduction section, can be refuted for the future trend of the city. The graphical and statistical analyzes carried out in the seminar paper have confirmed an decreasing trend of crimes in the future. Moreover, due the lack in data, the prophet package was an excellent alternative to the classical forecast package. It gives the possibility to deal with missing data and treats outlier very well. Further, based on the illustrated shiny application the seminar paper provides a wide overview about the current situation and the future outline of Chicago. Thus, it is a good tool to interactive manipulate the data and interpret further questions through the entirely visualizations. One point that lead to difficulties was the dataset, which was mainly filled with characteristic values, rather than numeric. This did not allow to perform classic regression analysis or forecasts using the classical ARIMA model. Therefore, the bulk of this work has fallen on the visualization of the data and the prediction through the prophet package.

A References

[1] Facebooks Core Data Science Team. prophet diagnostics. `https://facebook.github.io/prophet/docs/diagnostics.html`. Accessed: 20-02-2019.

[2] Harvey Andrew C. and Shephard Neil. Structural time series models. *Elsevier*, 11: 261–302, 1993.

[3] Papachristos Andrew V. 48 years of crime in chicago: A descriptive analysis of serious crime trends from 1965 to 2013. *ISPS WORKING PAPER*, pages 1–20, 2013.

[4] Shaw Clifford R. and McKay Henry D. Juvenile delinquency and urban areas. *Chicago: University of Chicago Press*, 1942.

[5] Griffiths Elizabeth and Chavez Jorge M. Communities street guns and homicide trajectories in chicago, 1980-1995: Merging methods for examining homicide trends across space and time. *Criminology*, 42(4):941–978, 2004.

[6] Silver Eric and Miller Lisa L. Sources of informal social control in chicago neighborhoods. *Criminology*, 42(3):551–584, 2004.

[7] Wojciechowski J., Hopkins AM., and Upton RN. Interactive pharmacometric applications using r and the shiny package. *CPT: Pharmacometrics & Systems Pharmacology*, 4(3):146–159, 2015.

[8] A. Ryan Jeffrey and Ulrich Joshua M. xts: Extensible time series. (1):1–22, 2008.

[9] Gardner Jr. and Everette S. Exponential smoothing: The state of the art. *Journal of Forecasting*, 4(1):1–28, 1985.

[10] Ignacio Medina, David Montaner, Joaquín Tárraga, and Joaquín Dopazo. Prophet, a web-based tool for class prediction using microarray data. *Bioinformatics*, 23(3): 390–391, 2007.

[11] Brockwell Peter J and Davis Richard A. *Introduction to time series and forecasting*. Switzerland Springer, third edition edition, 2016.

[12] Hyndman Rob J. and Koehler Anne B. Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4):679–688, 2006.

[13] Taylor Sean J. and Letham Benjamin. Forecasting at scale. *PeerJ Preprints*, 5: 1–25, 2017.

[14] Kim Gye Soo. Forecasting steem price prediction using r. *International Journal of Trend in Research and Development*, 5:325–329, 2018.

[15] Hastie Trevor and Tibshirani Robert. Generalized additive models: Some applications. *Journal of the American Statistical Association*, 82(398):371–386, 1987.

[16] Harry Willbach. The trend of crime in chicago. *Journal of Criminal Law and Criminology (1931-1951)*, 31(6):720–727, 1941.

B Appendix

Crimes and Arrests

```
Year12_16 <- cc[cc$Year2 %in% c("2012", "2013", "2014", "2015", "2016"),]
groupByDate <- na.omit(Year12_16) %>% group_by(Date2) %>% summarise(Total = n())
timeSeries <- xts(groupByDate$Total, order.by = groupByDate$Date2)
```

```
output$tsCrimeArrest <- renderHighchart({

hchart(timeSeries, name = "Crimes") %>%
hc_exporting(enabled = TRUE, filename = "TS_Crimes_Arrest") %>%
hc_add_series(arrestTimeSeries, name = "Arrests") %>%
hc_add_theme(hc_theme_smpl()) %>%
hc_title(text = "Time Series plot of Chicago Crimes") %>%
hc_subtitle(text = "(2012 - 2016)") %>%
hc_legend(enabled = TRUE, align = "center")
})
```

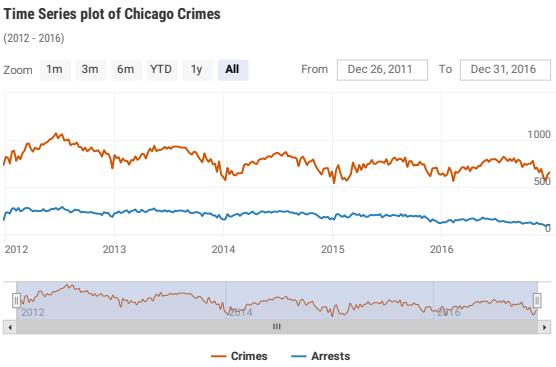

Figure 6: Time series of Crimes and Arrests.

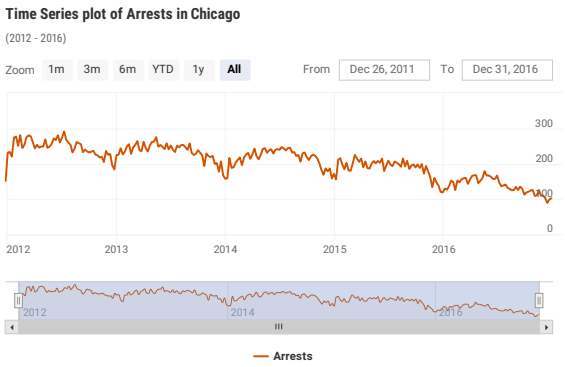

Figure 7: Time Series of Arrests.

Crime Types by Year

Crime type analysis

```
ctypeAnalysis <- cc[cc$Primary.Type == input$ctypeCrimeType,] %>% group_by(Year2) %>% ↩
    summarise(Total = n()) %>% filter(as.numeric(levels(Year2))[Year2] >= ↩
    input$ctypeDate[1] & as.numeric(levels(Year2))[Year2] <= input$ctypeDate[2])

   hchart(ctypeAnalysis %>% na.omit(), "column", hcaes(x = Year2, y = Total, color = ↩
       Total)) %>%
   hc_exporting(enabled = TRUE, filename = paste(input$ctypeCrimeType, "by_Year", sep = ↩
       "_")) %>%
   hc_title(text = paste("Crime Type by Year",input$ctypeCrimeType, sep = ": ")) %>%
   hc_subtitle(text = paste(input$ctypeDate[1], input$ctypeDate[2], sep = " - ")) %>%
   hc_xAxis(title = list(text = "Year")) %>%
   hc_yAxis(title = list(text = "Crimes")) %>%
   hc_colorAxis(stops = color_stops(n = 10, colors = c("#d98880", "#85c1e9", "#82e0aa"))) %>%
   hc_add_theme(hc_theme_smpl()) %>%
   hc_legend(enabled = FALSE)
```

Top 20 crimes analysis

```
crimetypeAnalysis <- cc %>% group_by(Primary.Type) %>% summarise(Total = n()) %>% ↩
    arrange(desc(Total))

   hchart(crimetypeAnalysis[1:20,], "column", hcaes(x = Primary.Type, y = Total, color = ↩
       Total)) %>%
   hc_exporting(enabled = TRUE, filename = "Top_20_Crime_Types") %>%
```

```
   hc_title(text = "Top 20 Crime Types") %>%
   hc_subtitle(text = "(2001 - 2016)") %>%
   hc_xAxis(title = list(text = "Type"), labels = list(rotation = -90)) %>%
   hc_yAxis(title = list(text = "Crimes")) %>%
   hc_colorAxis(stops = color_stops(n = 10, colors = c("#d98880", "#85c1e9", "#82e0aa"))) %>%
   hc_add_theme(hc_theme_smpl()) %>%
   hc_legend(enabled = FALSE)
```

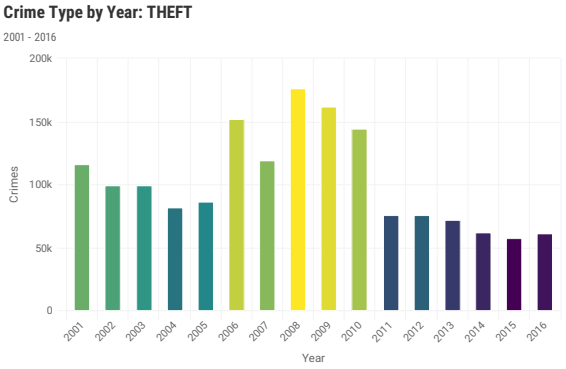

Figure 8: Crimes of the type Theft by year.

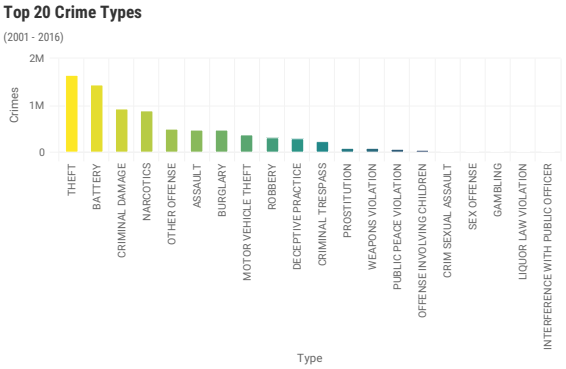

Figure 9: Top 20 crime types.

Crime Locations by Year

Crime location analysis

```
locAnalysis <- cc[cc$Location.Description == input$locLocation,] %>% group_by(Year2) %>% ↩
    summarise(Total = n()) %>% filter(as.numeric(levels(Year2))[Year2] >= input$locDate[1] ↩
    & as.numeric(levels(Year2))[Year2] <= input$locDate[2])

   hchart(locAnalysis %>% na.omit(), "column", hcaes(x = Year2, y = Total, color = Total)) %>%
   hc_exporting(enabled = TRUE, filename = paste(input$locLocation, "by_Year", sep = "_")) %>%
   hc_title(text = paste("Crime Locations by Year",input$locLocation, sep = ": ")) %>%
   hc_subtitle(text = paste(input$locDate[1], input$locDate[2], sep = " - ")) %>%
   hc_xAxis(title = list(text = "Year")) %>%
   hc_yAxis(title = list(text = "Crimes")) %>%
   hc_colorAxis(stops = color_stops(n = 10, colors = c("#d98880", "#85c1e9", "#82e0aa"))) %>%
   hc_add_theme(hc_theme_smpl()) %>%
   hc_legend(enabled = FALSE)
```

Top 20 location analysis

```
locationAnalysis <- cc %>% group_by(Location.Description) %>% summarise(Total n()) %>% ↩
    arrange(desc(Total))

   hchart(locationAnalysis[1:20,], "column", hcaes(x = Location.Description, y = Total, ↩
       color = Total)) %>%
   hc_exporting(enabled = TRUE, filename = "Top_20_Locations") %>%
   hc_title(text = "Top 20 Locations with most Crimes") %>%
   hc_subtitle(text = "(2001 - 2016)") %>%
   hc_xAxis(title = list(text = "Location"), labels = list(rotation = -90)) %>%
   hc_yAxis(title = list(text = "Crimes")) %>%
   hc_colorAxis(stops = color_stops(n = 10, colors = c("#d98880", "#85c1e9", "#82e0aa"))) %>%
   hc_add_theme(hc_theme_smpl()) %>%
   hc_legend(enabled = FALSE)
```

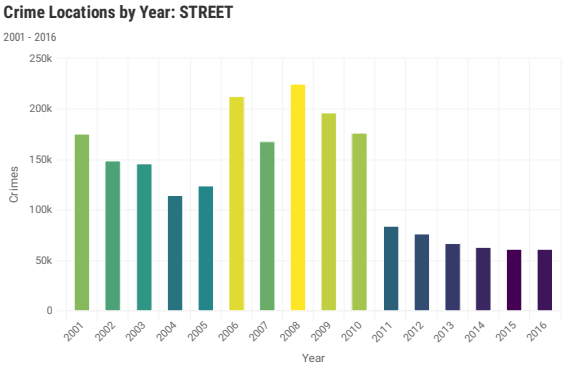

Figure 10: Crimes occurred on the Street by year.

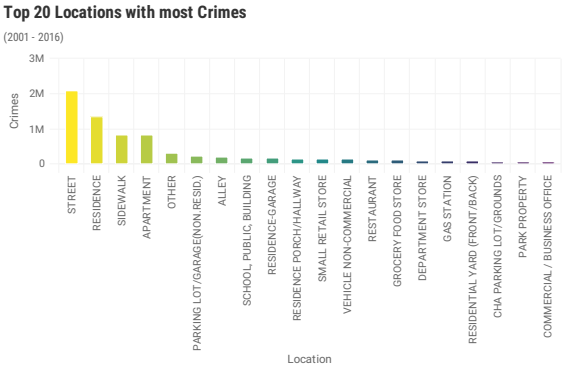

Figure 11: Top 20 crime locations.

Map

1. Create Map

```
output$map = renderLeaflet({
leaflet() %>% addProviderTiles(providers$Esri.WorldStreetMap) %>%
setView(lng = -87.623177, lat = 41.881832, zoom=11)
})
```

2. Filter Input

```
reactMap = reactive({
cc %>% filter(Primary.Type %in% input$mapCrimeType & Location.Description
%in% input$mapLocation & Year2 %in% cbind(input$mapYear[1],input$mapYear[2]))
})
```

3. Add Layers

```
observe({
proxy = leafletProxy("map", data = reactMap()) %>% clearMarkers() %>%
clearMarkerClusters() %>% addCircleMarkers(clusterOptions = markerClusterOptions(),
lng =~ Longitude, lat =~ Latitude, radius = 5, group = 'Cluster',
popup =~ paste('<b><font color="Black">', 'Crime Information', '</font></b><br/>',
'Crime Type:', Primary.Type,'<br/>', 'Date:', Date,'<br/>', #'Time:', Time,'<br/>',
'Location:', Location.Description,'<br/>', 'Block:', Block, '<br/>', 'Arrest:',
Arrest, '<br/>'))
})
```

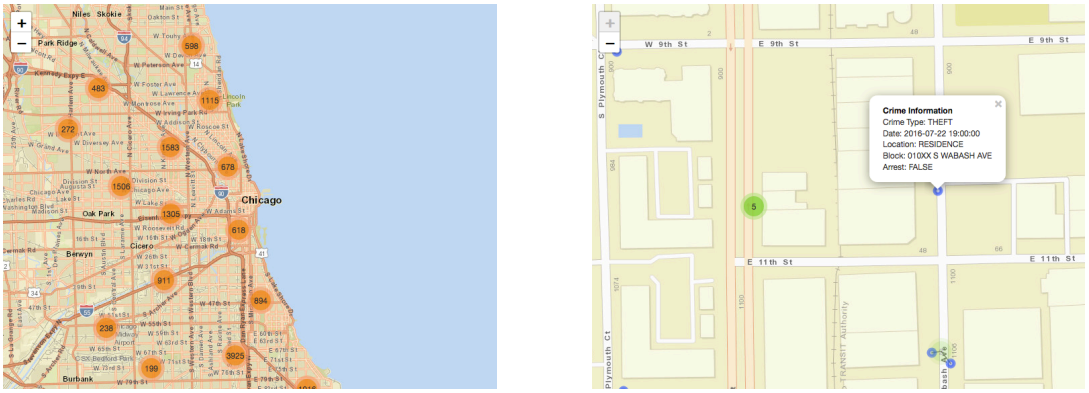

Figure 12: Map of Chicago City in normal and zoomed-in perspective.

Heatmap

1. Create Heatmap

```
output$heatmap = renderLeaflet({
leaflet() %>%
addProviderTiles(providers$CartoDB.DarkMatter) %>%
setView(lng = -87.6105, lat = 41.8947, zoom=11)
})
```

2. Add Heat

```
observe({
proxy = leafletProxy("heatmap", data = reactHeatmap) %>%
removeWebGLHeatmap(layerId = "hm") %>%
addWebGLHeatmap(layerId = "hm", data = reactHeatmap(),
lng =~ Longitude, lat =~ Latitude, size=120)
})
```

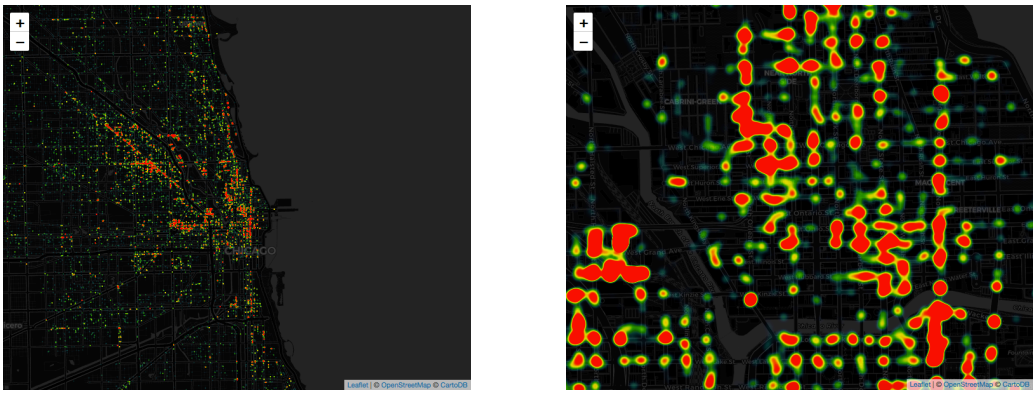

Figure 13: Heatmap of Chicago City in normal and zoomed-in perspective.