Monetorizer

Analysis of operation data

M A R C U S  B E R G M A N

KTH Computer Science and Communication

Master of Science Thesis
Stockholm, Sweden 2011
Monetorizer

Analysis of operation data

M A R C U S B E R G M A N

Master’s Thesis in Computer Science (30 ECTS credits) at the School of Computer Science and Engineering Royal Institute of Technology year 2011
Supervisors at CSC were Henrik Eriksson and Ann Bengtson
Examiner was Stefan Arnborg

TRITA-CSC-E 2011:119
ISRN-KTH/CSC/E--11/119--SE
ISSN-1653-5715

Royal Institute of Technology
School of Computer Science and Communication

KTH CSC
SE-100 44 Stockholm, Sweden

URL: www.kth.se/csc
ABSTRACT

This thesis presents a model for analyzing time series data. The time series data demonstrate the existence of seasonal patterns. This fact is used in the analysis by aggregating the data according to the seasonal patterns. From the aggregated data, trends are observed. The analysis conducted is requested by a company in the advertising industry. They seek to develop a tool to look at trends and indicate when data deviate from normal values. This thesis lays as a basis for the development of the tool sought.

The theory behind the model formed is discussed in the thesis. This includes some theory behind time series, pattern fitting and forecasting. For a better understanding of the model, the procedure is discussed. Apart from the input data, a number of variables which are tunable in order best forecasting. A major part of the analysis is to determine the best variable set of a number of different input data and variable set permutations.

The conclusion of the analysis conducted is that some variable sets are clearly favored compared to others. However, more work is needed in order to indicate when trend deviation occurs.

Monetorizer
Operationsdataanalys

SAMMANFATTNING


Slutsatsen från analysen är att vissa variabelkombinationer är klart bättre än andra. Dock behövs det utföras mer arbete för att trendavvikelser ska kunna upptäckas.
When it comes to writing the master’s thesis, most people I have talked to tell me that it has been the worst part in their struggle for a master’s degree. In a way, I can agree. It has been a rather long and from time to time hard walk. On the other hand I think the last six months could have been a lot worse than they have been. This is thanks to a number of people that have, in different ways, helped and pushed me along the way.

To Erik Hammarbäck and Therese Bergman for sharing thoughts and ideas and for their motivation from start to finish. May you finish your reports soon!

To my mother, Kerstin Bergman, for all those times at our summer house taking care of everything else for me to focus on my writing.

To my father, Henrik Bergman, for not being too nosy and pushy and for helping me with writing the abstract.

To my friends not yet having completed their degree, even though they should have. You have given me an extra push.

The final and foremost appreciation I want to give to Henrik Erikson for the never ending enthusiasm he has given me. He has an uncanny ability to turn defeat into success and this has given me energy to continue with this project.
1 INTRODUCTION ........................................................................................................... 1
  1.1 PURPOSE .......................................................................................................... 2
  1.2 OUTLINE OF REPORT ..................................................................................... 2

2 BACKGROUND ........................................................................................................ 3
  2.1 USER PERSPECTIVE ....................................................................................... 4
  2.2 THE AD PLAYER ............................................................................................ 5
  2.3 THE PLATFORM ............................................................................................... 7
  2.4 PLATFORM SUPERVISION ............................................................................. 10
  2.5 THE DATA AVAILABLE .................................................................................. 11

3 THEORY .................................................................................................................. 12
  3.1 TIME SERIES .................................................................................................. 12
  3.2 PATTERNS ........................................................................................................ 12
  3.3 PATTERN FITTING .......................................................................................... 15
  3.4 FORECASTING .................................................................................................. 17
  3.5 DATA MINING .................................................................................................. 19

4 PROCEDURE ............................................................................................................. 21
  4.1 SETTING THE GOAL ....................................................................................... 21
    4.1.1 The operational supervision model .......................................................... 21
    4.1.2 The automated checkup model ................................................................. 22
  4.2 OBTAINING SAMPLE DATA ........................................................................... 22
    4.2.1 Data structure .......................................................................................... 23
    4.2.2 Data extraction ......................................................................................... 27
    4.2.3 Site selection ............................................................................................ 29
  4.3 DEVELOPED TOOLS ....................................................................................... 30
  4.4 VISUALIZING SAMPLE DATA ......................................................................... 31
    4.4.1 Plain data Visualization .......................................................................... 32
    4.4.2 Daily / weekly data visualization ............................................................. 34
    4.4.3 Weekly 3D data visualization ................................................................ 35
  4.5 DATA ANALYSIS .............................................................................................. 36
    4.5.1 Model setup .............................................................................................. 37
    4.5.2 Variable setup .......................................................................................... 39
    4.5.3 Result set .................................................................................................. 40
    4.5.4 Result aggregation ................................................................................... 40

5 RESULTS .................................................................................................................. 42
  5.1 RESULT ANALYSIS ROUND I ....................................................................... 42
  5.2 RESULT ANALYSIS ROUND II ...................................................................... 44
  5.3 TAINTED RESULTS .......................................................................................... 45
1 INTRODUCTION

This report concerns data analysis in a certain company for academic purpose. But before going into specifics, let us take a look at the problem in the most general setting.

Information is a keyword today in business, science and politics, to mention a few. Society is flooded with information. We have more information than we can manage to understand and interpret. This is all thanks to computers and the Information Technology era. What do we do with all this information? We store it and use it when it is needed. But instead of looking at every single bit of information, we aggregate the information and process it, or mine data, as it is called. By doing this, we aggregate large data quantities to a smaller, more comprehensible amount of data we can interpret and use in business, science or politics. Our project fits into this broad picture.

The goal of this master’s thesis is to find and analyze trends on time series data. The trend analysis is intended to tell if latest acquired data follow the trends of the past. If the latest data do not, the deviation could indicate a problem that is needed to be taken care of. The data could refer to the number of people entering or exiting a subway station, how many light bulbs are sold at a convenience store or how many words an author writes. A crucial part when recording this type of information is to store it in time series. This means that the data are grouped to uniform time intervals. An easy example of this would be the light bulbs sold at a convenience store. A good resolution of the time series would perhaps be per day. The store owner can tell when light bulb restocking is needed most. For the author writing words, a good resolution could be per hour or per minute. From this information, the author can measure how fast he/she writes. The choice of resolution is based on what type of information is sought and what is available.

The title of the project, Monetorizer, is a word play based on the words ‘monetize’ and ‘monitor’.

Figure 1: Showing a graphical representation of sample data used. The graph shows the hour of the day along the x-axis, consecutive weeks along the y-axis and impression count on the z-axis. See data visualization in section 4.4.
1.1 PURPOSE

The data used in this project is gathered by VideoPlaza that works with online advertisement with emphasis on video advertisement. VideoPlaza provides the means for Web Video publishers to produce profit from showing advertisements in their Web Video players. VideoPlaza has a number of publishers as customers and records vast amount of data from each of the publishers' viewers. The data used for this report reflect how many successful video advertisements have been shown per hour. The result sought by VideoPlaza is an indication if their service seems to be working as intended for a publisher.

1.2 OUTLINE OF REPORT

After giving the background and explaining the data available, we will give a short introduction to the theory and practice of forecasting and data mining. We will discuss some procedures and analyzing methods that are appropriate in our context and perform an analysis of a large quantity of sample data. The results are presented and discussed. The report finishes with the conclusions gathered from the results and the project.
2 BACKGROUND

VideoPlaza provides ad serving and video advertising service to companies streaming video on the Internet. It was founded in Stockholm 2007 and has been a steadily growing company since then. At the time of writing, the company has approximately 30 customers (publishers) and automated operation surveillance is becoming increasingly important. In practice, what VideoPlaza does, is to provide the means to show advertisements and commercial banners in Web Video players. A Web Video player is a content frame located on a web page that can show video content. The best example is YouTube\textsuperscript{1}, where people from all around the world can watch content other people have uploaded. YouTube’s Web Video player is the actual part of the web page that shows the videos.

![YouTube Web Video Player](https://i.imgur.com/3.png)

**Figure 2:** Showing an example of YouTube webpage with its Web Video Player. The rest of the web page has been toned out to put focus on the Web Video Player. It is usually based on flash technology. This means that the flash player must be downloaded and installed on the computer where the user wants to watch YouTube. Another technology that is growing is HTML5 which makes it possible to show videos on a web page without external players.

Other examples of Swedish Web Video players (WVPs) are SVTPlay\textsuperscript{2}, TV4Play\textsuperscript{3} and Aftonbladet WebTV\textsuperscript{4}. Publishers of online video content may need to generate income from their WVPs due to the maintenance that comes with offering this kind of service. One solution is to make the Web Video player available only for paying customers. Another solution is to show advertisement. As mentioned above, this is what VideoPlaza can offer.

\textsuperscript{1} http://www.youtube.com/ (13)
\textsuperscript{2} http://svtplay.se/ (14)
\textsuperscript{3} http://www.tv4play.se/ (17)
\textsuperscript{4} http://www.aftonbladet.se/webbtv/ (18)
2.1 USER PERSPECTIVE

To get an understanding of how the video advertisement is perceived by a normal person using a video advertisement enriched Web Video player a use case is presented, with a fictional TV channel and a fictional TV show.

Charlie was planning on seeing the very popular program “The world’s most dangerous cougars”. Unfortunately he could not make it home in time to see the show on TV. Luckily for him, Channel SixtyNein shows most of its content online for a week after the show has aired. This includes the show in question. After a long exhausting day at the coffee shop, Charlie gets home, turns on his computer and logs on to the Internet. After he has entered Channel SixtyNein’s WebTV page he clicks on a picture link to the show.

The browser window loads the web page containing the WVP. The first thing that is shown is a commercial for shampoo and then another commercial for the area code lottery. Apparently, Charlie can win a lot of money if he signs up. He sighs.

Finally, the show starts. It is 30 minutes long. After a couple of minutes, a banner pops up that Charlie removes by clicking on the cross in the top right corner. The show continues to run and after 14 minutes, a commercial break is shown in the form of one video ad. When the “The world’s most dangerous cougars” has ended, another video ad is show. Charlie is tired and heads off to bed after shutting down the computer. He is very environment aware.

**Figure 3:** A fictional web video player displaying the show "The world’s most dangerous cougars". The control panel, seen in the lower middle of the player, has three vertical marks above the progress bar. This indicates there will be three commercial breaks throughout the show.
In order for Charlie to be able to watch shows, as well as other video clips from Channel SixtyNein, Charlie had to install the Adobe Flash player which is a common demand on many web pages working with interactive content. Channel SixtyNein uses VideoPlaza for showing ads and commercials. This means that their Web Video Player developers must modify their player in order to include the VideoPlaza ad player plugin into the video player. Luckily, this is not something Charlie has to care about, except that he will be interrupted by commercials and ads from time to time.

2.2 THE AD PLAYER

The ad player is a plugin that is designed so that it can be integrated into WVPs. This is something each publisher must do in order to be able to utilize VideoPlaza’s ad player plugin. Visually, the ad player plugin can be described as a layer that is placed on top of the publishers WVP. When ads are to be shown, the ad layer is set to be visual. When the video content is to be shown, the ad layer is hidden. Apart from this, there are also command interaction between the WVP and the plugin. For example, before a video clip starts, a commercial is shown. The WVP is paused during the commercial and when it has ended, the ad player plugin sends a command to the WVP to play. To any user watching the WVP with the ad player, this interaction and layer showing/hiding is not perceived. The user will think the showing of the ads is something that is a part of the WVP, if this is being reflected over at all.

![Figure 4: The picture shows a visualization of how the ad player plugin is placed on top of a WVP. The ad player is not visible when a video clip is shown but when a commercial is to be shown, the ad player layer is shown.](image)

Where, when, what and how often a video ad or banner should be shown is something that the publisher decides. It is managed on VideoPlaza’s web page. There are many different parameters that can be set and this affects how the ad player behaves in the publishers WVP.

Every ad that is shown is registered by VideoPlaza. This is done for several reasons. The first and foremost is for the sake of payment. VideoPlaza takes a small charge for every ad shown. The publishers, in turn, take a charge from the advertising
companies for showing their ads. Another reason for registering is to be able to observe trends and produce forecasts. The information gathered by the ad player is the primary and only source of raw data in this project. The details regarding this information will be discussed more in detail in section 4.2.

The ad player can display ads in several ways. The most common ad type is the video commercial. It can be shown before, during and after a video clip. Sometimes one commercial is shown at a time and sometimes several are shown. These ad types are called preroll, midroll and postroll respectively. Preroll is a video ad that is shown before the video clip, a midroll is shown during a video clip (like a TV commercial break during a movie) and a postroll is shown after a video clip. Another way is to show ads are with the use of banners. They can appear at anytime during a video clip and are shown throughout the clip or until the user closes the banner by clicking on a cross in the top right corner of the banner. This ad type is called overlay. See Figure 5 for an example.

Figure 5: A YouTube clip where a banner is being displayed. This type of ad is referred to as overlay.

The ad player can display ads in several ways. The most common ad type is the video commercial. It can be shown before, during and after a video clip. Sometimes one commercial is shown at a time and sometimes several are shown. These ad types are called preroll, midroll and postroll respectively. Preroll is a video ad that is shown before the video clip, a midroll is shown during a video clip (like a TV commercial break during a movie) and a postroll is shown after a video clip. Another way is to show ads are with the use of banners. They can appear at anytime during a video clip and are shown throughout the clip or until the user closes the banner by clicking on a cross in the top right corner of the banner. This ad type is called overlay. See Figure 5 for an example.

Figure 6 shows a WVP with its timeline beneath. The video and content watched by the users are a mix of the requested video clip and video commercials and a banner that are being displayed on top of the video clip.
2.3 THE PLATFORM

There are many other aspects to consider when explaining how the ad player works. The platform with all its servers is explained in this subsection. There are a number of different servers at work in the VideoPlaza platform setup. Each server or server group is appointed to one or more specific tasks. Major parts of the software running in the servers are developed by VideoPlaza. However, this is out of scope for this report. What is not out of scope is the platform setup.

This subsection shows and explains the VideoPlaza’s platform configuration. Some parts of the configuration have been left out because they are not needed to explain the data flow in this project. The hardware is mainly composed of virtual Linux machines. It is not needed for the reader to know the details of the hardware configuration and therefore these details will be excluded.

Figure 7 shows a subset of the VideoPlaza platform overview. The arrows show the way the data travel along the platform architecture. Each server included in the data flow will be explained in the subsequent paragraphs. There are two servers
that are not included in the data flow shown in Figure 7. They will be mentioned in short in the last paragraphs of this subsection. To understand the data flow in detail, the different steps along the way are explained below.

1. A user is starting to watch a Video clip. The WVP loads the ad player plugin. The plugin sends a ticket request. A ticket request is a message containing information about what publisher is making the request, tags belonging to the video clip (used to identify what advertisement should be used to the video clip), among other things. An advertisement is called a creative.

The ticket request is sent to VideoPlaza. The first point of the VideoPlaza platform that comes in contact with the ticket request is the HAPProxy (1). HAPProxy stands for high availability proxy and is used to load balance traffic between several proxy servers. The HAPProxy load balances ticket requests (among other types of traffic) between the proxy servers that handle the information.

The proxy servers receive the ticket request. Based on the information from the ticket request, the proxy decides what creative(s) the ad player should show. Sometimes, the decision is to show no advertisement. The proxy servers get the advertisement selection information from the server in the VideoPlaza platform called core. New creative decision information is
periodically uploaded to the proxies. The inner working of how advertisement selection is made is out of scope for this document.

2. When the advertisement information decision is made, a ticket response is sent back to the plugin in the WVP. The information sent contains what type of creative(s) should be shown and what/how the creatives should be used.

3. The plugin now knows what creative(s) should be displayed. The next step is to get the media content. Since it often is video advertisements, the size of the creative is too big to be transmitted from VideoPlaza when considering the number of creatives shown at the same time. The bandwidth load on both media servers and network would be too high. Instead VideoPlaza is using a third party Content Distributor Network (CDN) that has cache points located all over the world. The ticket response includes an URL to the best suited cache point based on the IP address of the viewer. The plugin requests the creative from that cache point. The creative is streamed to the plugin and begins to play.

4. Throughout the duration of the creative, playback events are recorded and sent to VideoPlaza. These events are, among others, beginPlay, 25% shown, 50% shown, 75% shown, endPlay, etc. These events are called tracker requests. The tracker requests are sent to VideoPlaza in the same way as the ticket requests.

5. Every ticket request, ticket response and tracker request is recorded and stored in one or several ways. The primary purpose for this is to have a basis when billing the publishers. Apart from the primary purpose, the tracker requests are needed for operational supervision, statistics and forecasting. More than just aggregating and storing data, the trackers also broadcasts current ticket and tracker information within VideoPlaza. The broadcast selected information is refined and used as speed meter information for operational supervision. The data is visualized in real time at VideoPlaza to see the current status, in whole as well as per publisher. More details regarding operational supervision are given in the next subsection 2.4.
6. The requests and responses are buffered locally on the proxies. At regular intervals the buffers are flushed to the tracker servers. The tracker requests received from the proxies are buffered locally. Periodically the tracker buffers are flushed to Mongo database servers. The data is aggregated in hierarchical level per hour and used for statistics. The data structure will be discussed in detail in section 4.2.1.

The data received by the tracker are also saved to the index server. The data stored are index and used to produce forecasts for the publishers.

The data flow explained above is, as mentioned earlier, only a subset of the complete data flow but it is enough to understand the basics in order to grasp the mechanisms that this project is based on. For the curious reader, the two servers seen in half transparent appearance in Figure 7 are core and db. They will be explained in short but the information given here is not needed in order to understand this project.

Core has three functions. It functions as the interface to the SQL server (db) where all configuration for how creatives should be shown and as the interface to the Mongodb (mdb) to access statistics. Core periodically distributes information to the proxy servers. This is how the proxy servers know what to reply in the ticket responses to the ad players explained in data flow step 2 above. Also, core responds to a REST API that is used by both VideoPlaza applications and publishers that needs to access information in the SQL and mongodb servers. The REST API adds a structured interface to these databases as well as a security layer so only allowed information can be accessed.

Db is a MySQL database server. Its purpose is to explain the relations between sites, campaigns, goals and ads among other relational information. Each publisher is referred to as a site. Each site has a set of campaigns that contains a number of criteria that explains which ad should be shown when a specific video clip is shown. Each campaign has a number of goals explaining how many times an ad should be shown during a period of time.

2.4 PLATFORM SUPERVISION

The VideoPlaza’s system is today being automatically supervised by the IT infrastructure monitoring system Nagios (2) together with resource monitoring tool Munin (3). These tools can accommodate most of the standard needs when it comes to host and network monitoring. Among the many parameters being
monitored, several parameters are special for VideoPlaza. One of these special parameters is called *impressions*, the number of video ads shown. If the impression rate is registering as zero, Nagios will detect this and raise an alarm.

Up until now, the automated monitoring, combined with manual checkups for each publisher impression statistics, have been sufficient for VideoPlaza’s operation. The manual checkups have been done to verify that the publishers have an acceptable impression flow. For example, if the flow rate decreases too much it is possible a technical issue has arisen.

VideoPlaza provides a forecasting tool for its publishers. The forecasting tool is used to predict how the impression rate will be in the near future. The resolution of the forecast is *per hour* and the prognosis is based on the same hour of the same weekday of the last weeks. The resolution used has been chosen out of experience. It is detailed enough to notice trend changes during the day and rough enough to smooth out momentary impression spikes and drops.

As the publisher base expands, it is becoming too time consuming for personnel to perform the manual checkups for all publishers. An alternative solution is needed.

### 2.5 THE DATA AVAILABLE

As mentioned earlier, VideoPlaza logs data. All data generated from ticket requests/responses and tracker requests are both logged and aggregated into more refined and usable information. Each one contains information such as time stamp, source IP address, geographic resolution and site-id (an id identifying which publisher the request/response comes from). Some information is unique to the different types of messages sent to and from VideoPlaza. Regarding the ticket request, the unique information in this message type are publisher-defined tags belonging to the video clip. For every ticket response, a number of ad ids are being included. The tracker response messages sent from the ad player contains much information, among others what type of ad is being displayed, is the tracker event registering an actual creative being display or was the ad player only registering a possible creative display. See section 4.2.1 for more detailed information about the tracker requests and its contained information.

For the curious reader, the average data flow per day to and from VideoPlaza is 100 MBit and 20 MBit respectively, where most of the traffic is generated from the tickets and trackers. All this data are called raw data and are dumped to log servers. All raw data are kept for safety reasons, should it be needed to extract information from the past. The total amount of log data generated per day is around 200 GB. When the logs are compressed they shrink to a tenth the original size.
3 THEORY

This section is dedicated to the theory behind the methods used for time series analysis and forecasting. The theory section is divided into subsections explaining the theory.

When dealing with large amounts of data, it is important to consider what type of data is used. In this case, the first thing to note is that the data are represented in such a way that it is called time series data. How is this known? For one thing, it can be seen when inspecting the data structure (see section 4.2.1). However, from a scientific point of view, Makridakis et al. (4) define time series in the following way:

\[ \text{Time series} = \text{An ordered sequence of values of a variable observed at equally spaced time intervals is referred to as a time series.} \]

3.1 TIME SERIES

The way time series data are structured makes it exemplary to be mathematically analyzed. The data follow a structure where the data values are tightly coupled with time. For an analysis to be performed without time aspect, this kind of data is not suited for that kind of task. Then another type of source data is needed. The different methods of analysis may be divided into two classes, when the analysis is performed with time in mind. The two classes are frequency-domain methods and time-domain methods (5). In our case, both types of methods were possible.

Frequency-domain methods focus on the properties for each given frequency, for example how well the data lies within each given frequency over a band of frequencies, phase shift information, etc. Methods included in this class are wavelet analysis and spectral analysis, where the Fourier transform is a well known method with a wide variety of applications.

Time-domain methods are applied to a time series as a whole and do not focus on individual frequencies. Methods included in this class are auto-correlation and cross-correlation.

In this study, the frequency-domain analysis is not of interest. There is no direct gain to understand the intricacies of how specific frequencies effect the overall time series. Time-domain analysis, however, provides useful information in this study.

3.2 PATTERNS

For time series there are always patterns that can be observed. To discover these patterns, the first thing to do is to visualize the data. This can easiest be done by graphing the data. Usually this is done with a time plot, which is a two dimensional
line graph with the time on the independent axis and the value in the dependent axis, as seen in Figure 12. Even though the description does not give away any information about what the data represent, patterns can be detected.

![Figure 12](image_url): Example of a time plot from time series data extracted from site 1 (see Appendix III). The time span is from the first of January to the last of January 2010.

One obvious pattern is that the graph goes up and down from day to day. It can be reasoned (with a fair amount of certainty) that the peaks occur in the days and the valleys occur at night. This would suggest the data correlate to human activity somehow. Apart from the daily pattern, the peaks themselves follow some sort of a pattern. However, without more information about what the data represent, it is hard to draw any conclusion as to what the pattern relates to.

To figure out the best way to analyze time series, it can be good to classify different patterns. Makridakis et al. (4) have distinguished four types of time series data patterns. Naturally, different patterns can coexist in the same time series.

1. **Horizontal pattern**
   
   When the data values fluctuate around a certain mean value, a horizontal pattern exists. The mean value is called the stationary point. Figure 13 show an example of this.

   ![Figure 13](image_url): A data plot where the data follows a horizontal pattern. The middle line is the stationary point of the data. The data is taken from Figure 12, looking at each day at 04:00.

2. **Seasonal pattern**

   When increases and decreases of the data values follow seasonal factors, such as the month, the day or the year, a seasonal pattern exists. An example is shown in Figure 12. We can clearly see that a daily seasonal pattern exists.
3. Cyclical pattern

When data values exhibit rises and falls with no apparent fixed periods, a cyclical pattern still exists. The main differences between this pattern and seasonal patterns are that the later has a fixed period length, whereas the former often vary in period length. The cyclic period length is also often longer than the seasonal and the magnitude of a cycle is usually more variable than that of the seasonal. An example is national monetary inflation/deflation changes. There is no easily predictable pattern in when inflation turns into deflation and vice versa but there is a pattern.

![Cyclical pattern graph](image)

*Figure 14: An observation of a cyclical pattern. The data is taken from site 7 (see Appendix III) with a data density of 2 weeks, to smooth out daily seasonal patterns.*

In the context of this project, we can display an example of a cyclical pattern. The pattern can be observed while observing the sample data that has data since a couple of years back. In the case above, in Figure 14, the cyclical pattern has a frequency of two per year. The peaks appear in the middle of spring and fall. The valleys appear around New Year and midsummer. The sample data are extracted from site 7, one of 15 sites used in this project. To make the graph cleaner, one data point per every second week is used. Read more about the sample data in section 4.2.

4. Trend pattern

When data values show a steadily long-term increase, or decrease, it is said that a trend pattern can be observed. A good example of this is when a public limited company’s share price is increasing in value over a period of time.

In Figure 15 a trend pattern can be observed. This graph shows sample data from site 8 with a reduced resolution to one data point per week. The graph shows a steady increase in its impression count.
The different patterns can, separate or in combination, give a clue to how to tackle time series data when trying to create forecasts.

### 3.3 PATTERN FITTING

Let’s say we are to analyze some five consecutive data points. Each data point has been recorded at 03:00 five Mondays in a row. We want to know if it is possible to say that the fifth data point can be estimated with the use of the first four data points. The first and perhaps most obvious way to do this numerically is to take an average from the first four points and compare it with the fifth data point to see if they correspond. See Figure 16. In some cases this would turn out to be true, in some cases not. It all depends on the data. If a sixth data point was added to the series (Figure 17), one way to change the model would be to calculate the average of the five first data points and compare it to the sixth. This would mean that no matter how long the data series becomes, all data points will always affect the result. This could be a disadvantage. Another way to tackle the addition of a new data point is to exclude the first (oldest) data point from the series and therefore let the model be influenced only by the latest data points. This method is
called moving average\textsuperscript{5}. In the case above, the last example the average was slightly closer than the example with six data points. This could be merely a coincidence, however.

\begin{center}
\includegraphics[width=\textwidth]{figure19.png}
\end{center}

\textbf{Figure 19:} Example of a graph where the data shows an increasing trend. A moving average (gray color) and a least square of the first order (light red color) are overlaid.

A moving average might sometimes not prove a good estimate of what is to come, as seen by the example in Figure 19 (following the same data as for the three examples above). The deep red horizontal line shows the moving average. An increasing trend can be observed from the data. A moving average would never take a trend into account. In order to take this into account another technique is needed; the least square method. It can fit a first degree polynomial line as closely together to a number of data points as possible. The result is seen as the light red color in the figure above.

\begin{center}
\includegraphics[width=\textwidth]{figure20.png}
\end{center}

\textbf{Figure 20:} Showing an increasing trend slowing down and beginning to decrease. The three fitting curves are least square fittings. The gray line a moving average, the light red line is of the first degree and the green curve is of the second degree.

To continue the discussion, what if the increasing trend is decreasing and is turning into a decreasing trend? The first degree polynomial line will not take this into account. In Figure 20 exactly this occurs. The light red line continues out above the graph. To take this change of trend into consideration, a polynomial of the second degree (green curve) is introduced and fits almost perfectly into the new data point.

By using higher degree polynomial a curve can be fitted perfectly too many data points but the predictions usually deteriorate. As seen in section X this has been tried and the result was unsatisfactory.

\begin{footnotesize}
\textsuperscript{5} http://en.wikibooks.org/wiki/Statistics/Summary/Averages/Moving_Average (12)
\end{footnotesize}
3.4 FORECASTING

Forecasting is a process that is used in many aspects of our lives. It is an essential process for many companies with highly diverse lines of business. It is used for weather forecasting, earthquake and tsunami prediction. Forecasting is a tool used to reduce company expenses as well as to save lives. Makridakis et al. (4) defines forecasting as the following:

“Forecasting is the prediction of values of a variable based on known past value of that variable or other related variables. Forecasts also may be based on expert judgments, which in turn are based on historical data and experience.”

As mentioned in the previous sections, it is good to graphically inspect data to get a feel for it. A data plot can give valuable information about the overall characteristics of the data. When it comes to the small differences and the actual forecasting, however, numerical analysis is needed.

A numerical summary of a data set can quantify it with a single value. This number is a statistic. A single data set, as will be used in this project, is called univariate data. When working with a pair of data sets, these are called bivariate and when having multiple data sets coupled together, they are called polyvariate.

Table 1: Showing the statistical formulae discussed.

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>( \bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i )</td>
</tr>
<tr>
<td>Mean Absolute Deviation</td>
<td>( \text{MAD} = \frac{1}{n} \sum_{i=1}^{n}</td>
</tr>
<tr>
<td>Mean Squared Deviation</td>
<td>( \text{MSD} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \bar{y})^2 )</td>
</tr>
<tr>
<td>Variance</td>
<td>( \sigma^2 = \frac{1}{n-1} \sum_{i=1}^{n} (y_i - \bar{y})^2 )</td>
</tr>
<tr>
<td>Standard Deviation</td>
<td>( \sigma = \sqrt{\sigma^2} = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (y_i - \bar{y})^2} )</td>
</tr>
</tbody>
</table>

When working with univariate data, mean and standard deviation are the most common numerical summaries performed. However, there are a couple more
statistics that is used. Table 1 gives a summary of the statistics used in this project. Each is explained in more detail below.

Mean, or average, is probably the most common and used numerical summary in everyday life. It is calculated by adding all values in the data set and the sum is divided by the size of the data set. Another very common numerical summary (not used in this project) is the median which is a selection of the middle observation in the data set.

Mean absolute deviation, or MAD, is a method not used very often but still good to use to find the mean deviation from the average. By extracting the absolute value from each deviation from the average \((y_i - \bar{Y})\), this summary is possible. Without the absolute value operation, the value would always result in zero. Basically it would be like subtracting the average from the average.

Mean squared deviation (MSD) and variance \((\sigma^2)\) are closely related and gives (without being picky) the same result. The small difference can be observed in Table 1. The reason as to why the variance is calculated by dividing the squared deviation from mean with n-1 as opposed to only n is because this leads to some desirable mathematical properties which are not discussed in this report. The variance has another important function. It is the calculating basis for standard deviation \((\sigma)\). Standard deviation is, just as MAD, a measure of spread. They measure (again, without being too picky) the average deviation from the mean value \(\bar{Y}\). In general, two thirds of some univariate data lie within one standard deviation from the main and 95% of all data points in that univariate data lie within two standard deviations.

With the knowledge of numerical summaries, the next step in forecasting is to produce output that predict future data points based on already known historical data. The method(s) used in this report have already been introduced in the previous section. However, they will be discussed some more here. The first method is the moving average, or moving mean, where a fixed number of values closest to the value to predict, are extracted from historical univariate data. See Figure 16 and Figure 18 in the previous section for a visual example. A mean value is calculated from this data set and the result is the predicted value. This method is very simple and does not take seasonal changes, cyclic patterns or trends into account.

The second method presented and used in this project is the least square method (LSM). It is a numerical method that fits univariate data to a polynomial of defined degree. The working principle is to solve an over-determined system of equations. Since an over-determined system of equations cannot be solved, the LSM finds a good approximation. See Figure 19 and Figure 20 for an example. A good method to solve the system of equations when the polynomial degree is not fixed is with the help of linear algebra. This is explained by Anton, Rorres (6) and Gerd Eriksson (7).
The result of a LSM is, as explained, a polynomial function of a degree determined when the least square matrix is set up. One thing that has to be considered when using the LSM is how to compute the deviation from mean \((y_i - \bar{y})\). For the all numerical summaries using this expression must replace it with \((y_i - f(x))\) where \(f(x)\) is the result of the polynomial function with \(x\) being the time sought.

### 3.5 DATA MINING

When a large amount of data exists and it needs to be understood and analyzed, this is often referred to as data mining. Michael J. A. and Gordon S. Linoff have written several books (8) (9) in this area. They define data mining as such:

*Data mining is the process of exploration and analysis, by automated or semi-automated means, of large quantities of data in order to discover meaningful patterns and rules.*

In this project, large quantities of data have been produced and with the help of data mining and statistics, the data size has been reduced to a manageable size. The section 4.5 - Data will discuss the practicalities about the data mining for this project. In this section data mining will be presented in more general terms.

Data mining can be directed or undirected. Directed data mining aims to build a model that can describe one defined variable based on all the data available. Undirected data mining seeks to find relationships between several variables. This is described in Mastering Data Mining (9). Directed data mining is divided into three activities, classification, estimation and prediction. Classification aims towards labeling (classifying) information, e.g. is a time series showing an overall increasing or decreasing trend. Estimation tries to quantify information. Following the previous example; how much the overall trend increase or decrease? Prediction is arguably not by itself a third and separate activity. It is more combination of the two other activities in where the difference lies within emphasis. Classification or estimation it is in fact a prediction but without the emphasis on if it actually is correct or not. When speaking about prediction, the emphasis lies the actual prediction, as in, was the classification of estimation correct? This can only be answered when the answer becomes available.

When setting out to come up with a model with the use of data mining techniques, the choice of input data is very important. We want to include exactly those data that have an impact on the target variable (the variable we want to predict). Some good advice is given in Mastering Data Mining (9). We cite the following two suggestions.
Consideration 1: Does the past predict the future?

This first consideration might seem obvious but it is still an important one. Often the past is a good prediction for the future but not always. The stock exchange is a good example where the past cannot predict the future. There, the correct model might be a random walk.

Consideration 2: Is the data available?

When setting up a predictive model it is crucial to think about the input data. It has to be available, not only in theory (“This data should not be hard to get...”) but in practice. This is especially important when a model demands input from several points of interest, e.g. several corporations. If the data are not available the model becomes useless.
4 PROCEDURE

This section is describing how the practical part of this master’s thesis has been carried out. The section is divided up into the steps according to work progress order. Each subsection explains its intention, procedure and a brief reflection of the work conducted.

4.1 SETTING THE GOAL

The first order of business is to establish the purpose of the master’s thesis, in detail. Since the project is carried out at a company, it is important to establish an ongoing dialog so that the work is progressing to their liking as well as to the satisfaction of the main purpose, the master’s thesis.

Before this master’s thesis was chosen and accepted, a very brief and not very detailed proposition had been made. The master’s thesis proposition was that an automated supervision system was needed. The intention was to supervise VideoPlaza’s core business, which is to provide their publishers with creatives, according to the policy of their own choosing. The requested function was to check the number of creatives per time unit (impression rate). There was already supervision of a static nature, meaning if a publisher’s impression rate were to drop below a certain absolute value, possibly zero, an alarm would be triggered. The supervision functionality asked for was to react to “not normal” impression rates.

The supervision was to be a function of a statistics analysis based on earlier recorded values. In what way this was to be done was not specified. Time series analysis was suggested as a good methodology to consider.

4.1.1 THE OPERATIONAL SUPERVISION MODEL

Even though the explanation was brief and not very detailed, the proposition seemed straight forward enough. However, the lack of clear definitions and explanation of the purpose in detail made the work lead into a side track that was not intended from VideoPlaza.

The original understanding of the purpose of an operational supervision model was to monitor the changes in impression rate on a fine granularity, in the magnitude of minutes or so. To come up with a statistics model that worked with live data, live data were needed in order to create it. As the investigation continued and it became more clear what had to be done to achieve this, data had to be recorded so
that it could be used in the statistics model creation later on. A quite extensive work was carried out in order to achieve this. The details of these proceedings are omitted since they have not been used in the final model. After some time the involved parties of this project, sat down and discussed the project. The goal was changed in order to better suit VideoPlaza’s needs.

### 4.1.2 THE AUTOMATED CHECKUP MODEL

The revised model would be an automated checkup for the Client Relations Department. The CR Department manually has to check its publishers’ impression rates for possible trouble. Either the publisher could have misconfigured the way creatives should be shown or there could be some serious problem such as software bugs or hardware failure causing the impression rates to behave abnormal. As VideoPlaza expands and gains more publishers, the focus on each publisher from the CR Department diminishes. This means that problems can occur without VideoPlaza noticing. Instead of VideoPlaza notifying its publishers of possible misconfiguration or other trouble it has become more the other way around, the publishers sometimes contact VideoPlaza to ask if there is anything wrong. VideoPlaza wants to be able to detect these problems before its publishers.

Realizing the practical aspect of the revised model also gave benefits. The impression rate was to be in the magnitude of x number of updates per day. This meant that already existing data structures could be used in the project. Recorded data measurements already existed in VideoPlaza’s MongoDB instances (see Figure 7 in section 2.3), measurements that would be ideal for my work. At this stage the goal was determined and decided to be final.

### 4.2 OBTAINING SAMPLE DATA

Obtaining sample data to work with has been important in order to understand the data, see the patterns and trends to be able to analyze and come up with a model that fits live data. The sample data is therefore live data that has been recorded and saved from the past. With the past live data, analysis could be performed on a massive scale. When working with such great amounts of sample data, this means that many variations, which exist in live data, can be taken into consideration and be taken into account when producing the model. As it turns out, data with the “correct” granularity does indeed exist since the start of VideoPlaza’s business. Live data have been recorded with an hourly frequency ever since they had their first customer, back in 2008.
The only problem with the sample data are that it does not indicate when a problem actually has occurred. For the analysis and model development, this is a big problem since it can only be assumed when a problem has occurred.

### 4.2.1 DATA STRUCTURE

Referring back to Figure 11 in section 2.3, it was shown that VideoPlaza uses a number of different persistent databases and data servers for their data storage. Presenting a short description of MongoDB from its own and Wikipedia webpage:

*MongoDB (from "humongous") is a scalable, high-performance, open source, document-oriented database. Written in C++. The database is document-oriented so it manages collections of JSON-like documents. Many applications can thus model data in a more natural way, as data can be nested in complex hierarchies and still be queryable and indexable. (10) (11)*

The structure of the part of interest is presented below. Following the data structure, a detailed explanation is given in order to understand every part’s details and function.

<table>
<thead>
<tr>
<th>Table 2: A structure diagram of the data extracted from Mdb.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data structure</td>
</tr>
<tr>
<td>vp</td>
</tr>
<tr>
<td>site</td>
</tr>
<tr>
<td>_id</td>
</tr>
<tr>
<td>key</td>
</tr>
<tr>
<td>u</td>
</tr>
<tr>
<td>id</td>
</tr>
<tr>
<td>e</td>
</tr>
<tr>
<td>t</td>
</tr>
<tr>
<td>p</td>
</tr>
<tr>
<td>date</td>
</tr>
<tr>
<td>modified</td>
</tr>
<tr>
<td>timeseries</td>
</tr>
<tr>
<td>&quot;01&quot;</td>
</tr>
<tr>
<td>&quot;00&quot;</td>
</tr>
<tr>
<td>&quot;01&quot;</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>&quot;23&quot;</td>
</tr>
</tbody>
</table>
The database \texttt{vp} is the database that contains almost all information needed to get the data for a publisher. As mentioned earlier, the \texttt{mdb} is structured in documents. All documents of this structure are located in \texttt{sites} and contain the objects \texttt{\_id}, \texttt{key}, \texttt{modified}, \texttt{timeseries} and \texttt{total}. \texttt{Site} is a reference to a publisher (or a test site etc) and is identified by the \texttt{site.key.id} and will be discussed in more detail later in this section.

![Figure 21: Showing the user interface of MongoDB. A client is connected to a VideoPlaza MongoDB instance. The user specifies to use the database 'vp'. Then one (random) mdb document from the 'site' collection is requested and displayed. To reduce the number of paged needed to display the entire document, the time series data have been chopped into two small pieces. Also, the '_id' and 'key.id' values have been chopped off.](image-url)

```
MongoDB shell version: 1.8.1
connecting to: test
> use vp
switched to db vp
> db.site.findOne();
{
   "id" : ObjectId("4c969..."),
   "key" : {
      "u" : "NOT UNIQUE",
      "id" : "a752f235-...",
      "e" : 8,
      "t" : "PREROLL",
      "p" : false,
      "date" : "2010-04"
   },
   "modified" : "2010-10-10 09:37",
   "timeseries" : {
      "01" : {
         "00" : 23,
         "01" : 13,
         "02" : 8,
         ...
         "23" : 47
      },
      "02" : {
         "00" : 33,
         "01" : 7,
         ...
         ...
      },
      "total" : 20581
   }
}``

As the key object node and timeseries object node are more complex than the rest of the object nodes, they are explained in more detail below. The object _id is an mdb internal identification that every document gets automatically when they are created. The modified object is a key mapped to a string value and carries a date time value in the format yyyy-MM-dd hh:mm (for instance: 2011-12-31 23:59). Every time the document is changed, the modified object is updated. The total object is a key mapped to an integer value which is a sum of all impressions in the document.

The Object “key” distinguishes each site document from each other with certain properties. These properties are:

- u – uniqueness (out of scope for this document)
- id – site identification
- e – event type
- t – ad type
- p – possible impression or real impression
- date – year-month key

Site identification or “id” identifies which site this document belongs to. A site is normally a reference to a publisher. Every time a publisher generates an impression, the knowledge of this impression is passed through the VideoPlaza platform structure as explained in section 2.1 and matched to one site document by this id and the rest of the key entries. The id value is of type string and is a mixture of five lower case alpha numerical groups delimited by dashes. To give an example, this site id belongs to the publisher that been used in all the publisher related examples in this text: 069873bf-fafa-4732-97c5-76484d077a73. The one and only piece of information the database does not contain is how to map a key.id with a site. This information was extracted from an internal VideoPlaza web page.

Event type or “e” represents the type of event that has occurred when a tracker request is send from the plugin. A number of different events are recorded but the only event type that matters when it comes to impressions, the event 0 (zero) is of importance.
Ad type or “t” represents what type of ad / creative that has been shown. There are a number of different predefined types that can be assigned:

- **MIDROLL** – A video ad shown in the middle of a video clip
- **OVERLAY** – A banner being displayed sometime during the video clip.
- **POSTROLL** – A video ad shown after a video clip
- **PREROLL** – A video ad shown before a video clip.

When a publisher’s video clip is viewed, creatives can possibly be shown but do not have to. This is predetermined by the publishers when they setup the policies for how often and what creatives are to be shown. Each video clip can show several creatives, both of same type or different types. In other words, one video clip can contribute to several recorded impressions. For example, an online video production of “The world’s most dangerous cougars” on Channel SixtyNein is shown. Recall the use case in section 2.1 on page 4. Before the actual show is shown, 2 video ads are shown. After three minutes an overlay creative is shown. In the middle of the program, another video ad is shown. When the show is done yet another video ad is shown. To sum everything up, two PREROLLS, one OVERLAY, one MIDROLL and one POSTROLL have been shown; resulting in five impressions having been generated throughout the “The world’s most dangerous cougars” show.

Possible impression or “p” is a true/false flag that indicates if the impression is a real recorded impression (p is false) or the impression is a possible impression (p is true). A possible impression is recorded when the publisher has shown a video clip, regardless if a creative has been shown or not. For every video clip that creatives are shown in, the publisher sets up a number of slots. The slots are placed either before (preroll slot), after (postroll slot) or within (midroll slot or overlay slot). To each slot a number is set that tells how many creatives can be shown within this slot. If a slot have the capacity of one creative and a creative was indeed shown, both a possible and a real impression are recorded. If the creative is not shown, only a possible impression is recorded. If the slot capacity is two and only one creative is shown, one real and two possible impressions are generated. Taking this information into consideration, it is prudent to update the example from before with the “The world’s most dangerous cougars” show. It is more correct to say that the real impression count is five. The possible impression count could be higher. The reason for recording possible impressions is to produce forecasts in order to give an estimate to the publishers as to how many potential impressions will come in the near future.
The *timeseries* object is structured in tree formation where each branch under the *timeseries* object node references to a day. Each day branch is sub branched into 24 hour branches. Each day branch is labeled and references to a day in the year and month given in *site.key.date*. To give an example, if *site.key.date* is carrying the value “2011-07”, the *timeseries* object would contain 31 day branches labeled “01” to “31” since there are 31 days in July. Each day branch’s 24 hour branches ranges from “00” to “23”, one for each hour of the day. An integer value is mapped to each hour branch, keeping track of how many impressions that have been registered during that hour with regard to the *site.key*. From the example above, the total number of values that would exist in the *timeseries* object node would be 31*24 = 744. In other words, July month have 744 hours.

To conclude the description of the data structure, it should be pointed out that for an ordinary video clip shown by a publisher, one or many site documents will be updated. Looking back at the example given when explaining the key object node above, there were a total number of five real impressions recorded. They were of types PREROLL, OVERLAY, MIDROLL and POSTROLL. To match these impressions their respective site documents, the rest of the key entries need to be known. The uniqueness *u* is has the value “NOT_UNIQUE”, the site identification *id* has the value “069873bf-fafa-4732-97c5-76484d077a73”, the event *e* has the value 0, *p* is false and the date is set to “2011-07”. The only key value that differs is the ad type *t* resulting in four site documents will be updated. To that we also have a number of possible impressions to take into account. Let’s assume that all five ad types could possibly have been shown in the video clip. This means that another five site documents will be updated. All in all, a number of nine site documents will have been updated when someone have finished watching an episode of “The world’s most dangerous cougars”.

### 4.2.2 DATA EXTRACTION

In the previous section, the relevant part of VideoPlaza mdb structure was explained in detail. This section will explain how the data were extracted from the mdb server in order to obtain the sample data and work with it.

It was decided that a complete copy of the mdb was to be given to me, since then I could work with the mdb as I pleased, without having to access VideoPlaza’s platform every time the data were needed or be cautious about interrupting VideoPlaza operations.
The running environment for the local copy of the mdb instance was chosen to be on a Linux host. The installation was easy and after some minor trouble with getting the VideoPlaza mdb copy to work, everything was installed and running. There are two ways to access data without any need for external tools. Either it can be done by the mdb command line interface, or with the use of java scripts. Since there are much data to be extracted from the mdb, the second way was chosen.

A java script was created that could extract all the data needed. Due to either little knowledge or the limitations of java script and mdb a bash script was created in combination with the java script so that extraction of data could be made in an easy and controlled fashion. A detailed explanation to the inner workings of the java script and bash script files will not be made but the scripts are included as Appendix I and Appendix II.

Using the bash script is fairly simple: export_site [site name(s)]. A site name can be, for instance, site1. If the argument to export_site is all, then all sites from the database will be exported. The output is a comma separated file named by

![Figure 22: Showing site 1 export data in MS Excel. The data was extracted from a VideoPlaza MongoDB instance to a comma separated file and imported into MS Excel. This figure shows a selection of the imported file, having 133,766 rows including the header.](image)
the site name with an ending of .csv (Comma Separated Values). Each record in the file contains the following fields: *rowId, event, type, possible, date, count*. In other words, all fields from the mdb site key + the date/time (yyyy-MM-dd hh:00) and each hours impression count. The only site key omitted is *uniqueness*. Event should have been omitted as well since it is always zero but the java script was written before I was aware that only event zero was of interest.

By studying the bash script in Appendix II, the first couple of lines show two variables called SITES and IDS. These two array variables are the connection in linking a site name (e.g. site1) with a site id. In Appendix II, all real site names and ids have been removed and replaced with two phony ones, to give the reader an idea of how they are used. The total count of sites in the database is approximately 200, though not all are real customers. Many of the sites are for the use of test purpose.

When the data had been extracted into .csv files, it was easier to get an overview of the data as well as graphing it (see next subsection). The site data domain start dates varied from middle 2008 to spring 2011. Each site was given a fixed start date set somewhat later that each site’s earliest recorded date. This was done because the data recorded in the earliest times often include many zero values.

![Site 5](image)

*Figure 23:  Showing the earliest recorded values for site 5.*

There can be several reasons for having these zero values at the beginning of a site’s time series data. The most common reason for this is because the publisher needs to test the functionality on a small basis before starting the ad service in full.

### 4.2.3 SITE SELECTION

The MongoDB database that contains all recorded data contains almost 200 sites. Most of the sites are not usable since most sites are not real customers but test or development sites. Of the sites that are customers, the 15 most active sites were selected to be used as the sample data. The most active sites generate the highest
amounts of impressions. The impression rates from the selected sites have peaks ranging from 8,000 to 500,000 impressions per hour. The normal impression rates are considerably smaller. Each site is labeled Site1 to Site15. Site1 have already been used in most of the figures earlier in this report.

4.3 DEVELOPED TOOLS

In order to generate visual representations of the sample data as well as performing the numerical analysis, several tools have been developed. These tools have helped much throughout the project. For the most part, Java SE-1.6\(^6\) with the Java IDE Eclipse\(^7\) combined with JFreeChart\(^8\).

The java code foundation was created during the operational supervision model phase and was later adapted and modified to work with the data and structure change when the automated checkup model was decided to be the final one. A number of data import, data manipulation and data visualization were created along with a number of help classes. To the right here, all classes used are listed in a package divided view. The (default package) contains all actual programs used. Program.java has been used as an early test program to verify internal time series data structures and functionality. Program3D.java was developed as a continuation of Program.java with some interesting 3D visualization capabilities that has been used when creating the 3D graphs. Since JFreeChart does not support 3D plots in a way useful to visualize time series data, another open source library called jMathPlot\(^9\) was used for generating these graphs. See section 4.4.3. ProgramAnalyze.java analyzes the sample data and produces output, either to standard output, as text files or as graphs. More about the analysis can be read in section 4.5.

---

\(^{6}\) http://download.oracle.com/javase/6/docs/ (19)

\(^{7}\) http://www.eclipse.org/ (15)

\(^{8}\) http://www.jfree.org/jfreechart/ (16)

\(^{9}\) http://jmathtools.berlios.de/doku.php (20)
**ProgramBatch.java** is the program used for generating all 2D graph pictures. It works by passing a configuration file location as a command line argument to the program. The program interprets the configuration file and generates either a .png image based on the configuration file, or it opens an interactive java window displaying the graph. All the different command that exist in the configuration will not be mentioned in this report but it can be said that they are based on keys and values. Each row in the configuration file starts with a key (e.g. `source`) and is followed by the value (e.g. `C:\data\site1.csv`). To make the ProgramBatch.java easier to handle, GraphBatch, a simple user interface, was also created. This program was created in C#.NET, mainly because it takes less time to make small simple programs with user interface in .NET compared to java (writer’s opinion). GraphBatch was created to be as nonintrusive as possible and is basically a button that can be pressed (all configuration files are processed) or be right-clicked (a menu showing all configuration files are opened). From the menu each configuration file can be edited, see Figure 25, or generated. When a graph is being generated, the GraphBatch program simply passes the configuration file location as an argument to ProgramBatch and waits until the java program has completed its task.

**ProgramConf.java** generates configuration files for ProgramBatch.java when it is too demanding and time consuming to write each configuration file by hand.

### 4.4 VISUALIZING SAMPLE DATA

When working with data of this quantity, it can be very hard to understand data and grasp the patterns and variations in it. To circumvent this inability, it is wise to try and visualize the data in order to understand it. It can give a hint of how to tackle a problem at hand. When visualizing data, there are several aspects to
consider. One is perhaps the most obvious; how should the data be visualized? A common way of doing this is by graphing the data. Another way is with the use of video visualization but this will not be used in this report.

There are many ways to graph data. When it comes to time series, a good type of graph is the line graph, where the independent x axis represents time and the dependent y axis represents the corresponding value. In this case, dependent y axis represents impression count for every hour. Another point to consider is what tool should be used when visualizing the data. There were three choices. Either I could use the text editor built in chart diagrams, a powerful professional data tool like MATLAB or I could use a preexisting graph library written in the programming language Java. There are several pros and cons for each choice. The built in chart diagrams in Word (the text editor used writing the report) is too simple and too inaccurate for what needed to be used. MATLAB is indeed a good choice but my current MATLAB knowhow was not the best. I felt that the latter choice suited me the best. With the use of the Java IDE Eclipse\textsuperscript{10} combined with JFreeChart\textsuperscript{11}, a free licensed Java library for creating graphs, it was now possible to develop code to visualize the sample data available.

Before beginning the visualization phase of this project, a hint was given that the time series data seemed to have a strong weekly repetition. This was one milestone to consider when visualizing the sample data. However, it was still important to not rely solemnly on this fact when analyzing the sample data. If the sample data analysis led more strongly towards another direction, that was the path to follow.

4.4.1 PLAIN DATA VISUALIZATION

The first step in visualizing the sample data is to make a plain and simple time series chart of the data existing. Below the sites are visualized, one by one. To reduce the amount of data and focus on only one type of data, only the PREROLL creatives are to be analyzed. The PREROLL creatives, as explained earlier, are the video ads that are shown before a video clip and are the most commonly used creatives. Below is the chart for Site1 that spans from September 2009 to August 2011. Charts for all 15 sites can be found in Appendix III. There the reader can see the difference in impression rate, the length of the time series and more for each site.

\textsuperscript{10}http://www.eclipse.org/ (15)
\textsuperscript{11}http://www.jfree.org/jfreechart/ (16)
In Appendix III all site’s sample data are plotted. As can be seen, it is not very easy to visually inspect the data in detail. Figure 27 shows the plot for site one. In order to detect reoccurring patterns, a closer look at part of the data is needed. Figure 12 in section 3.2 shows data for the span over a month. This figure has a better resolution to understand the data pattern.

A good example of a reoccurring (seasonal) pattern was discovered while working with the operational supervision model (from section 4.1.1). Observe Figure 28. The graph shows 14 one day graphs overlaid. A daily pattern is observed.

From the looks of this one sample it seemed that the daily pattern really was a good start. Perhaps an even better seasonal pattern could be found by looking at the weekly pattern.
From the looks of Figure 28 and Figure 29 it seems that the weekly reoccurring pattern is more tightly coupled compared to the daily pattern. However, pure visual observation is not an acceptable method to base a conclusion on. A numerical analysis is needed.

4.4.2 DAILY / WEEKLY DATA VISUALIZATION

Before starting with the numerical analysis it was prudent to get an even better understanding for the data. As seen in the previous subsection, the sample data had a seasonal pattern with a frequency of both a day and a week. The next step was to visualize the sample data with the frequency of just one day or one week. The two examples of the result can be seen below.

There are more frequent data peaks/falls in the graph with a frequency per day (Figure 30) than in the graph with a frequency per week (Figure 31). Since each site has its own pattern traits, the graphs vary in characteristics. Some graphs are smoother, others are rougher. Site 7, shown in the figures above, exemplifies relative normal pattern trait, compared to the other 14 sites. Site 7 neither emits abnormal smoothness, nor extreme roughness.
One major problem with this kind of 2D visualization is that when one reduces the data frequency from per hour to per day / week, most of the data cannot be seen, if one does not generate many graphs. 24 graphs for per day frequency change and 168 graphs for per week frequency change. Doing so demands much time going through all the graphs and the sheer number of pictures would most likely be too many to make sense of as well. However, an alternative would be to use three dimensions to visualize the data instead.

### 4.4.3 WEEKLY 3D DATA VISUALIZATION

Practically, it was somewhat harder to generate the 3D graphs compared to the 2D graphs. This was because it demanded the use of another java plot library. JFreeChart does not (at time of writing) support 3D surface or line plots. To circumvent this problem, the library JMathPlot was used. The java interfaces of the two libraries differ some but not very much and it was easy enough to adapt to JMathPlot. However, it was not as straightforward to save 3D images as it is to save 2D images. Before saving a 3D image, a camera view must be decided since 3D plots must be projected onto a 2D image. Different sample data must be viewed from different angles in order to capture as much of the 3D data as possible. In practice, this demanded that each 3D graph to be generated had to be handpicked and saved manually.

One conclusion drawn from previous section 4.4.2 is that there is little use in continuing to produce graphs with per day frequency. For this part only graphs with per week frequency were to be produced. This made the development of the Program3D.java faster to complete since a less dynamic program was needed.

The first and at the time seemingly most natural way of generating the 3D plot, was to use a 3D mesh. An example can be seen in Figure 32. When watching the result, the instant conclusion was that it did not give a very good understanding of the data. The graph seems cloggy and hard to understand.

**Figure 32:** Site 9 Monday data visualized from 2011 week one to sixteen in the form of a 3D mesh.
One attribute making it cloggy was the mesh lines that exist in both width and depth, making it very hard for the viewer to understand how to interpret the graph. Another disrupting attribute is the mesh. It is half transparent, making folded areas darker. It is also difficult to know which lines, in the folded areas, represent which hours of the day.

The next and final idea that came to mind involves a 3D line plot. The result can be observed in Figure 33. This approach gives a more natural understanding for how the data points correlate to each other. To make the lines more self-explanatory, they are colored after what time of the day they represent. The black color represents night, the green color represents morning, the blue color represents day and the orange color represents evening. The result gives a good understanding of a big amount of data visualized in one graph.

4.5 DATA ANALYSIS

The previous section was dedicated to get a feel for the data from the 15 sample sites. In this section, the numerical analysis of the data takes place. The first step to be decided how the analysis should take place. What method(s) should be used? How are the methods tested? How are the results verified? These questions will be answered within the following paragraphs. The theory behind the analysis can be read about in chapter 3. The results are presented and discussed in chapter 5.

The sample data from the 15 sites contains a large amount of time series data. Before any numerical analysis was performed, the data was visualized. Since the database contains many different data series (see section 4.2.1), one was selected to be analyzed. The parameters for this series has the event type “PREROLL” and possible false. That means that the data series we are watching are all creatives shown before the video clip and that we only consider actual show creatives (and not all creatives that could have been shown).

To understand the number of data points to be analyzed Table 3 gives a summary regarding this. Each site has a start date which is when the first data was recorded.
All sites have data recorded until 2011-08-18 23:00 except for site 4 which stopped its service 2011-02-09 23:00. The column *Recorded* tells how many data points each site has stored. The next column, *All*, is the data point count with injected zero point values wherever a value is missing. Recall the last paragraphs from section 4.2.2 Data extraction as to why. Whenever an hour passes without any data is recorded the data is not written to the database. For the internal java data model to work, a continuous data series without any data gap is needed in order to work. The column *Analyzed* shows how many data values for each site that is to be included in the analysis.

**Table 3**: Displaying information for the sites that is being used in the analysis.

<table>
<thead>
<tr>
<th>Site</th>
<th>First recorded Date</th>
<th>Recorded</th>
<th>All</th>
<th>Analysis Start Date</th>
<th>Analyzed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2008-12-15 18:00</td>
<td>22912</td>
<td>23429</td>
<td>2009-01-11 00:00</td>
<td>22799</td>
</tr>
<tr>
<td>2</td>
<td>2011-03-01 07:00</td>
<td>3977</td>
<td>4096</td>
<td>2011-03-13 00:00</td>
<td>3815</td>
</tr>
<tr>
<td>3</td>
<td>2011-03-24 10:00</td>
<td>3541</td>
<td>3541</td>
<td>2011-03-24 00:00</td>
<td>3551</td>
</tr>
<tr>
<td>4</td>
<td>2009-02-05 12:00</td>
<td>8730</td>
<td>17628</td>
<td>2009-04-01 00:00</td>
<td>16321</td>
</tr>
<tr>
<td>5</td>
<td>2009-05-18 16:00</td>
<td>18403</td>
<td>19736</td>
<td>2009-06-01 00:00</td>
<td>19416</td>
</tr>
<tr>
<td>6</td>
<td>2010-02-10 12:00</td>
<td>12609</td>
<td>13307</td>
<td>2010-02-11 00:00</td>
<td>13295</td>
</tr>
<tr>
<td>7</td>
<td>2008-08-18 03:00</td>
<td>26052</td>
<td>26301</td>
<td>2008-08-19 00:00</td>
<td>26280</td>
</tr>
<tr>
<td>8</td>
<td>2010-05-07 13:00</td>
<td>6473</td>
<td>11243</td>
<td>2011-01-04 00:00</td>
<td>5447</td>
</tr>
<tr>
<td>9</td>
<td>2009-07-14 10:00</td>
<td>18344</td>
<td>18374</td>
<td>2009-07-15 00:00</td>
<td>18360</td>
</tr>
<tr>
<td>10</td>
<td>2010-12-02 10:00</td>
<td>5801</td>
<td>6229</td>
<td>2010-12-22 00:00</td>
<td>5759</td>
</tr>
<tr>
<td>11</td>
<td>2010-02-22 13:00</td>
<td>12990</td>
<td>13018</td>
<td>2010-02-23 00:00</td>
<td>13007</td>
</tr>
<tr>
<td>12</td>
<td>2010-04-19 08:00</td>
<td>9644</td>
<td>11680</td>
<td>2010-04-21 00:00</td>
<td>11640</td>
</tr>
<tr>
<td>13</td>
<td>2010-05-10 22:00</td>
<td>8491</td>
<td>11162</td>
<td>2010-09-03 00:00</td>
<td>8400</td>
</tr>
<tr>
<td>14</td>
<td>2010-11-08 14:00</td>
<td>4872</td>
<td>6801</td>
<td>2011-02-03 00:00</td>
<td>4727</td>
</tr>
<tr>
<td>15</td>
<td>2009-05-15 18:00</td>
<td>18641</td>
<td>19806</td>
<td>2009-05-18 00:00</td>
<td>19752</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>181480</td>
<td>206351</td>
<td></td>
<td>192569</td>
</tr>
</tbody>
</table>

4.5.1 MODEL SETUP

With the data range defined and ready to be analyzed, it is now time to present how the analysis was conducted. Apart from the sample data, a set of input variables are defined:

**Table 4** lists the three variables that are used in the analysis with a short description for each one.

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>Seasonality frequency.</td>
</tr>
<tr>
<td>N</td>
<td>The number of most current past valued to be compared with a value.</td>
</tr>
<tr>
<td>P</td>
<td>The polynomial degree(s) that should be used when performing the fitting model.</td>
</tr>
</tbody>
</table>
It is these three variables that tweak the prediction model. The model is fairly simple. The first step is to divide the data into the data frequency or seasonality. This would mean that a seasonality of one day results in 24 data sets, one for each hour. Was the seasonality of one week, 168 data sets are created. A simplified example can be observed in Figure 34.

Each of the seasonality data sets are then divided into subset. The size of the subset is defined by N. Each subset is used to compare with the data point next after the subset. In Figure 35, the data point being compared with subset 1 is the third last data point. For subset 2, the next last and for subset 3 the last data point. Even though it is possible to create a fourth sub set, which includes the last data point, it would not serve a purpose since there is no data point to compare the subset with.

![Figure 34](image)

**Figure 34:** Example showing how data is divided into three seasonality data sets. To make it easier to distinguish the data points from each other, the original data frequency have been reduced to 8 hour. On top of this the seasonality is set to one day, meaning there are three seasonality data sets.

![Figure 35](image)

**Figure 35:** Example showing a data set divided into subsets. The subsets are to be compared with the data point following the subset. The gray-tones of the subset outline correspond to the data points with the same tone of ring outline.

The two figures above have the same data input. In Figure 35, only the first seasonality dataset is show. In this dataset, three subsets are compared to three data points. If we were to include all data from Figure 34, this would undeniably lead to a total of nine subsets being compared to nine data points when the data input consists of 18 data points. Are we to change N from three to four in our example we can perform six comparisons and with a N set to five, three comparisons can be done. When including all results we have 9+6+3=18 comparisons done.
In the Theory chapter, pattern fitting was discussed (Section 3.3). This theory is not put into practice. For each subset generated, the least square method, LSM, is performed, to find fitting curves. How complex the fitting curve should be, is decided by the last variable \( P \) in Table 4. For the LSM to work, the subset must contain at least one data point more than the polynomial degree. In the example above, where \( N=3 \), it would be possible to use LSM in the data sets with degree 0, 1 and 2. Each result can then be compared with the data point to see how close the estimated point is to the real data point. With all nine subsets, we have 27 results to compare. To complete the number of result possible to get (\( N \) is 3, 4 or 5), the grand total number comparisons produced are \( 9*3 + 6*4 + 3*5 = 66 \).

The purpose of the model described is to derive a set of parameters that is good to use when analyzing data. If a publisher has a trend that is out of the ordinary, it should be detected.

### 4.5.2 VARIABLE SETUP

To get an understanding of the model, many variable sets are used in the model. Both variable sets that make sense and don’t make sense are being tested in the model, in order to see if the model itself is working or not. For each of the 15 sites, the following variable sets are used. \( N \) is ranging from 3 to 10. \( P \) is ranging from 0 to \( N-2 \) but at the most \( P = 5 \). The reason for not selecting a \( P \) higher than five is twofold. The higher \( P \), the quicker the predictions usually deteriorate. Also, in the case the results of the selected range of \( P \) indicate that a higher \( P \) is more suitable, a follow-up analysis, utilizing this knowledge, is to be conducted. The seasonality setup is listed in the table below.

<table>
<thead>
<tr>
<th>Table 5: Seasonality setup.</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Seasonality period</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hour</td>
<td>1, 2, 4, 8, 16</td>
<td></td>
</tr>
<tr>
<td>Day</td>
<td>1-6</td>
<td></td>
</tr>
<tr>
<td>Week</td>
<td>1-4</td>
<td></td>
</tr>
</tbody>
</table>

From the content in Table 5, say the seasonality period is “hour” and the period range is 2, two seasonality data sets are generated. One holding all date that has its hour even, the other set holding all dates with odd hours. Would the combination be “day” and 2, there would be 48 seasonal data sets, one for each hour for the two days.
The total number of seasonal frequencies used is 15. Together with N and P, which for each seasonal frequency have 33 different combinations, the variable set has a total of 495 variable combinations.

### 4.5.3 RESULT SET

When calculating each result set from each subset, mean, mad and standard deviation (\(\sigma\)) is calculated. A quote (\(Q\)) is also calculated to see how many standard deviations the predicted value (\(V_p\)) is from the real value (\(V_r\)). The formula:

\[
\frac{V_r - V_p}{\sigma} = Q
\]

This means that when \(Q\) equals 1.23 the real value is 1.23 standard deviations larger than the predicted value. Is \(Q\) negative, this means that the predicted value is larger than the real value. This result set is calculated for every P for every N for every seasonality frequency.

The result set produced for each site is significantly big. Each seasonal data set will produce a result set that is (set length – N) big for each P. For example, site 1 has 22799 data points in its total data set. Say we use this entire set, meaning the seasonal frequency is per hour. N is set to 5. The size of the result sets produced is 22795. Change the seasonal frequency to every second hour. We then get two data sets, 11399 and 11398 is lengths. The size of the result sets produced is 11395+11394=22789. Again, change the seasonal frequency to per day and the lengths’ of the data sets are 950 (all sets except one) and 949. The size of the result sets produced is 23*945 + 944=22679.

### 4.5.4 RESULT AGGREGATION

Since the result set is large, it is unpractical to go through all result data manually. It must be aggregated is some fashion. The results from subset are processed with the models explained in section 3.4. This means that mean, mad and standard deviation (\(\sigma\)) is calculated. These calculations were first performed only on \(Q\) but after some consideration, the calculations were performed also on the absolute difference (\(V_p - V_r\)) and the relative difference (\(V_p/V_r\)). These additional numbers are to help to understand the magnitude of the original results.

The result now generated is far less in numbers compared the first results. However, depending on the seasonality frequency, the data are still very large.
When having the seasonality frequency per day there are 24 results for every N and P. If the seasonality frequency is per week, this number rises to 680. It is still too large a number to inspect manually. To reduce this result set even more, all values calculated from the first set of aggregation (for each N and P) the mean value is calculated. To visualize this two step aggregation, the data in Figure 34 and Figure 35 is reused and presented in Figure 36.

![Figure 36](image)

Figure 36: Showing a visual representation of the original result sets produced as well as the two aggregated result sets.

Naturally, by aggregating the result, all variations that are important to study are lost. The aggregation is used only to get the general idea if a variable set is a good candidate for a closer inspection. The total result count from every site is now 495. These results are saved in 15 .csv files, one for each site. The next task is to analyze the results.
5 RESULTS

The aggregated results to go through are, no matter how aggregated, still a lot. It is also important to understand what the result mean. The main focus has been put at the standard deviation which, in more explainable terms, means how much dispersion the results have. Dispersions have been calculated from the absolute difference, the relative difference and the number of standard deviations the predicted value differed from the real value, Q. To find a good value, Q’s standard deviation (Qσ) should be as small as possible.

5.1 RESULT ANALYSIS ROUND I

By studying the smallest value of Qσ from each site gives an interesting result that can be seen in Table 6.

<table>
<thead>
<tr>
<th>site</th>
<th>Seasonal</th>
<th>Freq</th>
<th>N</th>
<th>P</th>
<th>ABS σ</th>
<th>REL σ</th>
<th>Q σ</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Hour</td>
<td>4</td>
<td>6</td>
<td>0</td>
<td>13906,22</td>
<td>8,82</td>
<td>1,28</td>
</tr>
<tr>
<td>2</td>
<td>Week</td>
<td>2</td>
<td>9</td>
<td>0</td>
<td>4055,52</td>
<td>5,15</td>
<td>0,35</td>
</tr>
<tr>
<td>3</td>
<td>Hour</td>
<td>4</td>
<td>6</td>
<td>0</td>
<td>5505,14</td>
<td>2,33</td>
<td>1,13</td>
</tr>
<tr>
<td>4</td>
<td>Week</td>
<td>3</td>
<td>10</td>
<td>0</td>
<td>588,08</td>
<td>5,84</td>
<td>1,89</td>
</tr>
<tr>
<td>5</td>
<td>Week</td>
<td>4</td>
<td>10</td>
<td>0</td>
<td>2567,66</td>
<td>9,82</td>
<td>1,4</td>
</tr>
<tr>
<td>6</td>
<td>Week</td>
<td>4</td>
<td>10</td>
<td>0</td>
<td>1022,05</td>
<td>1,39</td>
<td>1,25</td>
</tr>
<tr>
<td>7</td>
<td>Week</td>
<td>3</td>
<td>10</td>
<td>0</td>
<td>2789,97</td>
<td>2,5</td>
<td>1,32</td>
</tr>
<tr>
<td>8</td>
<td>Week</td>
<td>4</td>
<td>6</td>
<td>0</td>
<td>17940,98</td>
<td>0,29</td>
<td>0,77</td>
</tr>
<tr>
<td>9</td>
<td>Week</td>
<td>2</td>
<td>9</td>
<td>0</td>
<td>1087,24</td>
<td>6,5</td>
<td>1,54</td>
</tr>
<tr>
<td>10</td>
<td>Week</td>
<td>3</td>
<td>9</td>
<td>0</td>
<td>1363,35</td>
<td>1,76</td>
<td>0,41</td>
</tr>
<tr>
<td>11</td>
<td>Hour</td>
<td>2</td>
<td>10</td>
<td>0</td>
<td>849,66</td>
<td>10,74</td>
<td>1,46</td>
</tr>
<tr>
<td>12</td>
<td>Week</td>
<td>4</td>
<td>9</td>
<td>0</td>
<td>522,62</td>
<td>10,52</td>
<td>1,19</td>
</tr>
<tr>
<td>13</td>
<td>Week</td>
<td>4</td>
<td>10</td>
<td>0</td>
<td>1122,43</td>
<td>0,67</td>
<td>1,05</td>
</tr>
<tr>
<td>14</td>
<td>Week</td>
<td>2</td>
<td>10</td>
<td>0</td>
<td>6230,47</td>
<td>2,98</td>
<td>0,97</td>
</tr>
<tr>
<td>15</td>
<td>Week</td>
<td>2</td>
<td>8</td>
<td>0</td>
<td>1246,97</td>
<td>12,65</td>
<td>1,61</td>
</tr>
</tbody>
</table>

The results presented in the table above gives us some useful information. The seasonal frequencies favoring a good result are the upper weeks (2-4), this is not very unexpected. However, site one and eleven have every fourth respectively every other hour as their best choice. This is more unexpected and will be further discussed later in this chapter. N seems to range from the mid values to the upper values. The most definite piece of information is that P=0 in all cases. This will be kept on mind when continuing to discuss the analysis and results.
To get a more diverse result to study, the top ten results each from the absolute dispersion (ABS σ), the relative dispersion (REL σ) and the Q dispersion (Q σ) values were selected. An almost instant observation was made. The variations of variable P are shown in Table 7.

Table 7: The variations of P for the top 10 results for each of the categories ABS σ, REL σ and Q σ.

<table>
<thead>
<tr>
<th>P=0</th>
<th>P=1</th>
<th>P=2</th>
<th>P=3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>234</td>
<td>110</td>
<td>40</td>
</tr>
<tr>
<td>Percentage</td>
<td>60 %</td>
<td>28 %</td>
<td>10 %</td>
</tr>
</tbody>
</table>

P=0 is the most reoccurring value. In second place comes P=1. In third place comes P=2. P≥3 is practically nonexistent. It is perhaps not surprising that big values of P are excluded but what is surprising is that P=0 is most reoccurring. From this observation, all variable sets with P≥2 can be excluded. This reduces the continued result analysis.

By checking the occurrences of N, the same way as for P the conclusion to be drawn is that the longer N gives a better result. See Table 8.

Table 8: The variations of N for the top 10 results for each of the categories ABS σ, REL σ and Q σ.

<table>
<thead>
<tr>
<th>N=3</th>
<th>N=4</th>
<th>N=5</th>
<th>N=6</th>
<th>N=7</th>
<th>N=8</th>
<th>N=9</th>
<th>N=10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Count</td>
<td>37</td>
<td>32</td>
<td>32</td>
<td>40</td>
<td>53</td>
<td>56</td>
<td>70</td>
</tr>
<tr>
<td>Percentage</td>
<td>10 %</td>
<td>8 %</td>
<td>8 %</td>
<td>10 %</td>
<td>14 %</td>
<td>14 %</td>
<td>18 %</td>
</tr>
</tbody>
</table>

An interesting and unexpected result was discovered when looking at the occurrence of seasonal frequencies (SF). The most occurring SF is per hour. This is possibly because the trend hour to hour is fairly reoccurring. It should also be noted that a half of all occurrences P=1,2,3. This would support the assumption just stated. The second most reoccurring SF is seen per day, four times more than per week but fairly similar to per every other, third and fourth week. The results can be studied in Table 9.

Table 9: The variations of SF for the top 10 results for each of the categories ABS σ, REL σ and Q σ.

<table>
<thead>
<tr>
<th>Hour/ Day/ Week</th>
<th>H</th>
<th>H</th>
<th>H</th>
<th>H</th>
<th>D</th>
<th>D</th>
<th>D</th>
<th>D</th>
<th>W</th>
<th>W</th>
<th>W</th>
<th>W</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>16</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>Count</td>
<td>90</td>
<td>3</td>
<td>22</td>
<td>11</td>
<td>8</td>
<td>63</td>
<td>6</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>17</td>
</tr>
<tr>
<td>Percentage</td>
<td>23 %</td>
<td>1 %</td>
<td>6 %</td>
<td>3 %</td>
<td>2 %</td>
<td>16 %</td>
<td>2 %</td>
<td>1 %</td>
<td>1 %</td>
<td>1 %</td>
<td>4 %</td>
<td>12 %</td>
</tr>
</tbody>
</table>
What can be said about the result from the first round then? Well, perhaps not as much as hoped for. Some assumptions made in the beginning of the project seem not to be true. One example is the fact that the SF per week is far less occurring than per day. Another being that the most occurring P is zero. Naturally also that the SF per hour is the most occurring of all SFs. On the other hand we can eliminate the use of high order polynomials. We also see that the SFs per second, third (and so forth) per day can be eliminated. From common sense we can also eliminate all SFs per x day. We don’t want to observe value changes from hour to hour. We know it will vary throughout the day. One variable that is still difficult to tackle is N. Looking at the results in Table 8 the natural conclusion should be that a bigger N is favored as opposed to a smaller N. This is supposedly true but there could be a danger of using a too long N. Old data combined with a small P makes new trends hard to distinguish.

5.2 RESULT ANALYSIS ROUND II

The eliminations that can be done from round one leaves us with a smaller variable set when continuing with round two. N is now ranging from 3 to 9 with a two unit increase (to reduce the number of variable sets). P is now ranging from 0 to 2. The new seasonality setup is listed in Table 10.

<table>
<thead>
<tr>
<th>Seasonality period</th>
<th>Period range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Day</td>
<td>1</td>
</tr>
<tr>
<td>Week</td>
<td>1, 2, 3, 4</td>
</tr>
</tbody>
</table>

By going through all sites’ results from round two a clear winning variable set can be seen. The result from the smallest Q σ from every site is shown in Table 11. Some if the sites contained faulty data (marked with a star). This is either because the sample data are too short to be able to analyze with all variable sets or because the sample data contained too many zeros which makes dividing impossible. What can more be said about the results from round two is that per 2/3/4 week SF are favored compared to per week and per day SF. This fits with the observations made when the sample data was visually inspected. It seems that it is better to compare data every second to fourth week than every week. An interesting point to note is that a normal moving average is favored compared to line and curve fittings. This can probably be explained if some effort was put into it. However, that is unfortunately out of scope for this report.
Table 11: Showing the results from analysis round II.

<table>
<thead>
<tr>
<th>Site</th>
<th>1</th>
<th>2*</th>
<th>3*</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8*</th>
<th>9</th>
<th>10*</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14*</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>9</td>
<td>9</td>
<td>7</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>5</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>P</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

All results observed and discussed have been aggregated two times. Recall result set 3 in Figure 36 (where the aggregation steps were explained). To verify that the aggregated result actually is correct, checks were made with data from result set 2 from the same figure.

5.3 TAINTED RESULTS

When first working with LSM and the results were visualized, the results didn’t seem to be correct. This assumption turned out to be correct. The underlying cause to this problem was the data input. Since the x-axis is date-based, it was needed to be converted into numerical values, in this case the Java data type double. The most natural way to convert a date into a numerical value is to use Java’s internal data type, milliseconds since epoch (1970-01-01 00:00). The double data type have no problem in representing the converted values, the problem lies in how double represents numbers. Since one hour is a very insignificant change compared to the time since epoch. However, the double data type has no trouble representing each of the hours. The problem occurs when the LSM matrix is solved. The small significance in between the numbers causes the result matrix to be tainted. In some cases the matrix couldn’t even be solved since it became singular.

To counteract the insignificance problem all domain values were shifted towards zero by setting a reference date and subtracting that values from all dates before entered into the LSM. This improved the LSM results.

To get an understanding of this domain shift two pictures are presented. One, Figure 37, showing the correct LSM fitting, the other, Figure 38, is clearly tainted.
Figure 37: Showing a graph with sample data and correct curve fitting.

Figure 38: Showing a graph with sample data but tainted curve fitting.
6 CONCLUSION

One of the first assumptions that were made when setting up the automated checkup model was that the best set of variables should make the predicted value differ as little as possible from the real value. This meant that the sample data are assumed to be within acceptable fluctuations for the most part. One question that can be asked is the following one: Is this assumption a good one? The answer is twofold. On one hand no, it would be presumptuous to say so without even verify the assumption. On the other hand yes because the verification is not possible since the data, sadly enough, for making this verification does not exist. It has to be assumed that the data are ok for the most part since VideoPlaza is a growing organization that has satisfied customers for the most part.

Something to consider is how accurately the model actually should fit the data? From a scientific point of it is interesting to see how close one can get the model to forecast the data. On the other hand, from a business point of view this might not be something to strive towards. Take the scenario that there is a continuous decreasing trend for a publisher. Depending on the model variable set, different outputs would be generated from them. If the model was based only on a moving average it might be so that a remarkably large $Q_\sigma$ is calculated. Fair enough. What about a first or even second degree least squares method was used? This would mean that the model expects a decreasing trend and the forecasted value might be right at the new data point. From this, the model says: “Yes, this is an acceptable value. It’s right on target!” From a business point of view a decreasing market is something not desired. The models and model testing performed have resulted in numerical output. The result does not tell if the data analyzed is following the trend or not. It does not tell if the trend is positive, negative or neither. This is something that has to be defined in order to reach the real goal of this project.

Directed data mining was discussed in section 3.5. It contains three activities; classification, estimation and prediction. The numerical analysis in this project can be perhaps best associated with estimation. One could argue that prediction is a better fit since we are trying to predict. Perhaps there is no best answer. Perhaps the real reason for trying to label the data mining process is for the labeler to stop and think about what the purpose of the data mining really is. Are we trying to classify, estimate or predict the data output? In the end a classification is sought. Is the data indicating that something is wrong? Yes or no.
There are very many ways to tackle the problem in this project. Without trying all known ways, it is difficult to know which way is the best. The path chosen to tackle the data is probably not the best and hopefully not the worst. Even though it would be splendid to find the best model but that is not the main purpose of this project. The purpose is to look at the data, try out a model to see if it works. Evaluate it to see what have to be modified in order to make the model better or rather good enough. The aim for his project was to create a model with different variables sets to find the best suited variable set. When this was done a prototype was to be constructed to test the model in its intended environment. The latter part of this aim could not be completed due to lack of time. This is unfortunate. However, the prototype will certainly be created to test the model.

Things can always be improved and become better. It is easy to point these improvement potentials. Equally important to reflect on is what have been gained and what has been learned. One of the most important things, in this project, that have been gained is knowledge of the patterns of the sample data. The analysis has given an insight which can be put to use later in a follow-up project. This type of knowledge can easily be taken for granted and therefore neglected to reflect upon. Moreover, the model testing has given a good basis in forming a practical prototype. Last but far from the least, the tools developed for this project have a potential of coming to good use in the future.


function printSiteData(clientID)
{
  var q={"key.id":clientID,"key.e":0,"key.u":"NOT_UNIQUE"};
  var rowId=0;
  var c=db.site.find(q).sort({"key.t":1, "key.date":1});

  print("RowId,event,type,possible,date,count");

  while( c.hasNext() ) {
    var o = c.next();
    for( day in o.timeseries ) {
      for( h in o.timeseries[day] ) {
        var v = o.timeseries[day][h];
        print( rowId++ + "," + o.key.e + "," + o.key.t + "," + o.key.p + "," + o.key.date + "," + day + " " + h + ":00," + v);
      }
    }
  }
}
APPENDIX II

Script language: bash script
File name: export_site
File description: Bash script file is used to export time series data from mdb. See section 4.2.2.

#!/bin/bash

SITES=('site1' 'site2')
IDS=('e48a3bc8-09ce-2b35-8b8f-d82413ea07f6' '6181705a-401e-465e-b7c0-33caal6976b')

OUTPUT="/home/user/output/
JSTEMPLATE="/home/user/export_site.js"
TEMPFILE="tmp.js"

#####################
# Params:            1) SITE-ID  2) file-path
# Input/Output:      writeSite $1 $2
# Found/Not Found:
#
function writeSite()
{
  echo "SITE-ID: $1"
  echo "FILE-PATH: $2"

  cp -f "$JSTEMPLATE" "$TEMPFILE"
  echo "printSiteData("$1");" >> "$TEMPFILE"

  mongo --quiet vp "$TEMPFILE" > $2

  #rm -f "$TEMPFILE"
}

#####################
# Params:            1) site-name (according to $SITES)
# Input/Output:      returnString=$(getSiteID $siteName_var)
# Found/Not Found:   if [[ $? == 0 ]]; then echo "Found"; else echo "Not found"; fi
#
function getSiteID()
{
  site="$1"
  len=${#SITES[@]}

  for (( i=0; i < $len; i++)); do
    s="${SITES[$i]}"
    if [[ "$s" == "$site" ]]; then echo "$IDS[$i]"
  esac
}
return
fi
done
echo "[no_exist]"
return 1
}

if [[ "$1" == "" || "$1" == "-h" || "$1" == "--help" ]]; then
echo "Exports a mdb site or all mdb sites"
echo "Usage: $0 all"
echo "  Exports all sites to $OUTPUT"
echo ""
echo "$0 [list of sites]"
echo "  Exports specified sites to $OUTPUT"
echo ""
echo "$0 -h"
echo "  Show this help."
echo ""
else
for i in $@; do
  if [ "$i" == "all" ]; then
    len=${#SITES[@]}
    OUTPUTFILES="$OUTPUT*"
    rm -f $OUTPUTFILES
    for (( i=0; i < $len; i++ )); do
      s="${SITES[$i]}"
      id="${IDS[$i]}"
      file="$OUTPUT$s.csv"
      writeSite $id $file
    done
  else
    # Call function for doing the stuff
    id=$(getSiteID $i)

    if [[ $? == 0 ]]; then echo "Found: $id"; else echo "Not found"; fi

    file="$OUTPUT$i.csv"
    writeSite $id $file
  fi
done
fi
APPENDIX III

Description: This appendix shows a graphical representation of every site used in the report. The time span differs from site to site since they were not put in production at the same time. For more details about the site visualization, see section 4.4.1.