Case Study 1: Working with Edgar datasets: Wrangling, Pre-processing and exploratory data analysis

Team 4: Emily Strong and Raksha Kaverappa

Code Repository: <https://github.com/erstrong/INFO-7390-ADS-Fall-17-TeamNo.4/tree/master/Assignment%201>

**Problem 1: Data wrangling Edgar data from text files**

Docker Repository: <https://hub.docker.com/r/estrong/as1problem1>

AWS Bucket: <https://s3.console.aws.amazon.com/s3/buckets/akiai3qvj2wiihrzpvwa-output>

**Design and Implementation:**

With a given CIK and accession number we generate the URL to get that company's Filing Detail page by concatenating the Edgar archives url with the CIK, a shortened version of the accession number with hyphens removed, and the accession number followed by the index page suffix. The CIK and accession number are user-supplied values read from an ini config file. If either of them is invalid and thus the generated URL is invalid, the script throws the error "CIK or accession number is invalid" when trying to retrieve the page.

We then scrape the tables on that page into a Pandas data frame using the Pandas read\_html function. We verified that the target table is consistently the first table on that page on a sampling of Filing Detail pages for Dow Jones Industrial Average companies. We identify the 10-Q url based on the row with Type 10-Q and use a similar concatenation to retrieve the 10-Q report.

On the 10-Q report the tables are formatted inconsistently and though the Pandas read\_html function did scrape all the tables, there were problems with data types that led us to use Beautiful Soup instead. This required replacing newlines and no break spaces with empty strings to make the data more human-friendly, and iteratively appending table rows to a Pandas data frame. We then exported each table to a separate csv file that is zipped with a log file of each action performed by the script and uploaded to an AWS bucket named based on the user's key id plus a suffix. The AWK key id and secret key must be supplied by the user in the ini file. If either is invalid we retrieve the error returned by AWS, read the error code, and return "Invalid AWS key id" or "Invalid AWS secret key". Finally, we automated this process with Docker using an Anaconda 3 image and confirmed that the pipeline works for a sampling of other companies from the Dow Jones Industrial Average.

**Analysis:**

While the 10-Q URL scraping portion of our code is a success, the final csvs from the 10-Q tables contain irregular formatting including empty rows and text split over multiple rows, as well as columns lacking headers. Cleaning up these irregularities requires regex and we were not able to determine the correct pattern that would generalize across all 112 tables in the initial page we analyzed, and on testing our code with other 10-Q filings we found that the number of tables on a page is not consistent, so we couldn't determine appropriate regex on a per-table basis.

**Problem 2: Missing Data Analysis**

Docker Repository: Could not get image to run

AWS Bucket: [https://s3.console.aws.amazon.com/s3/buckets/akiai3qvj2wiihrzpvwa-outputproblem2](https://s3.console.aws.amazon.com/s3/buckets/akiai3qvj2wiihrzpvwa-output)

**Design and Implementation:**

For the second part of the case study, we are reading the URL for every month generated based on a year provided in an ini config file. We are reading the data from the zip-files instead of extracting them due to the large size of the csv log files. We have performed missing data analysis on every single file. The missing numeric values are assigned to 0. The missing string and object values are assigned “Unknown”. The missing file sizes are replaced with mean of all sizes. These missing value substitutions were chosen based on the Readme file that was provided with the log files.

We then perform summary metrics on the files. The summary metrics considered were:

* Identifying the file that was accessed the most for every month
* Frequencies of browsers that was used
* Histograms of hourly activity (included below)
* Identifying the top-5 CIK for every month

We observed anomalies in the file size. Outliers were identified after plotting the size in a box plot. The boxplots are included below.

**Analysis:**

Across the entire 1 year data set, one access number was the most commonly viewed file three months in a row: 0001209191-03-031123. This turns out to be a Form 4 filing (a document related to changes in ownership to prevent insider trading) for a company known at the time as I Flow Corp, regarding the shares of Jack Halperin against whom there was a 2016 SEC settlement for misconduct.

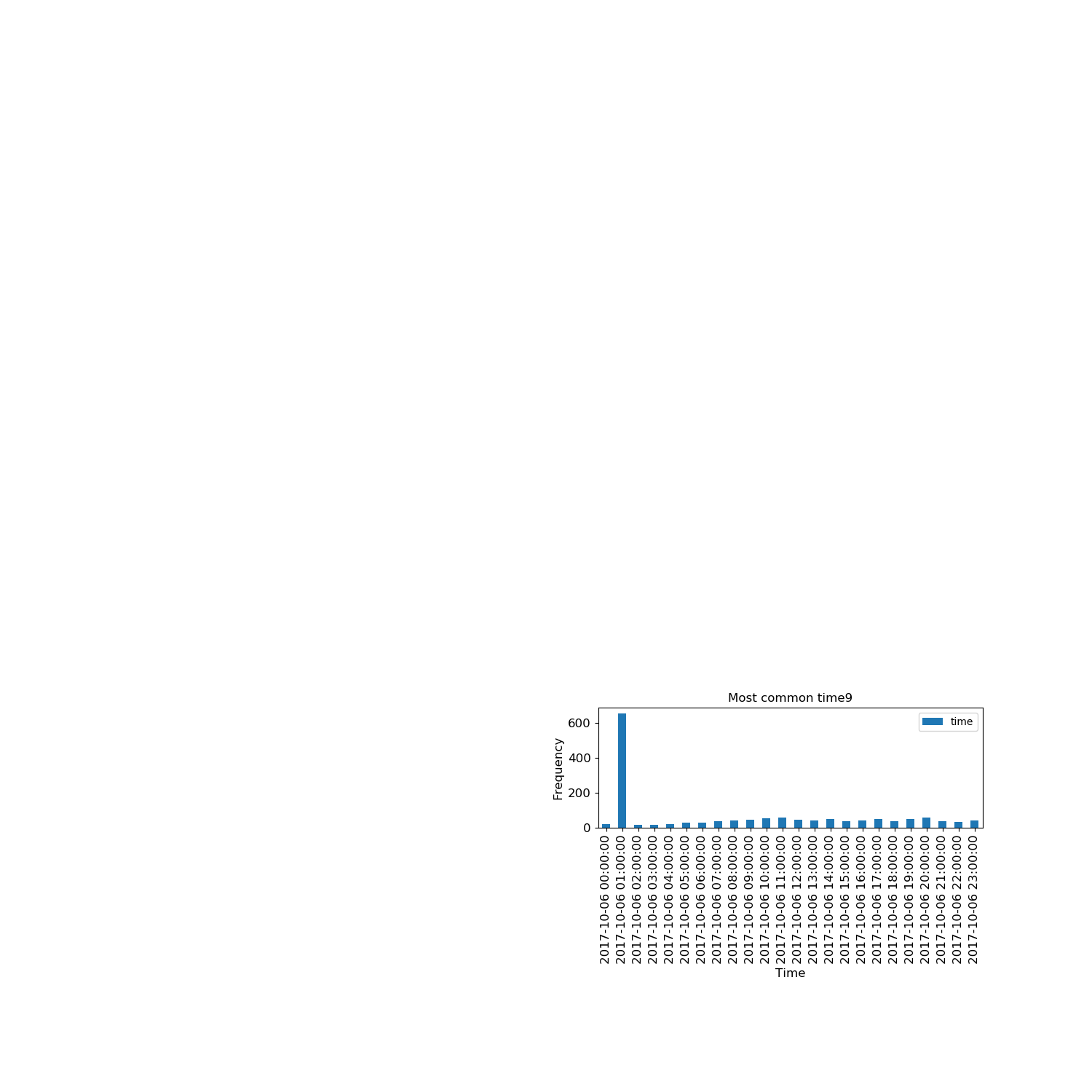
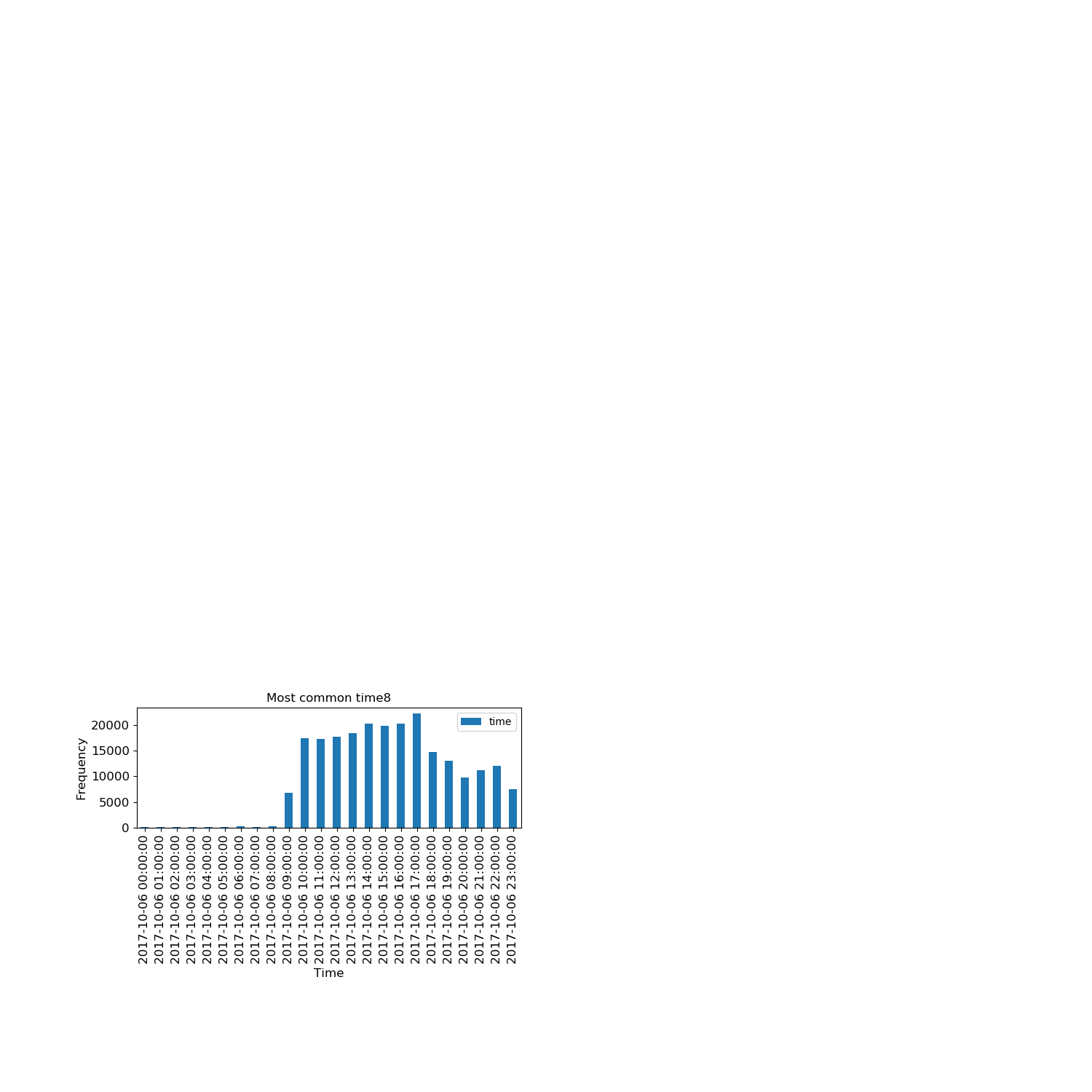
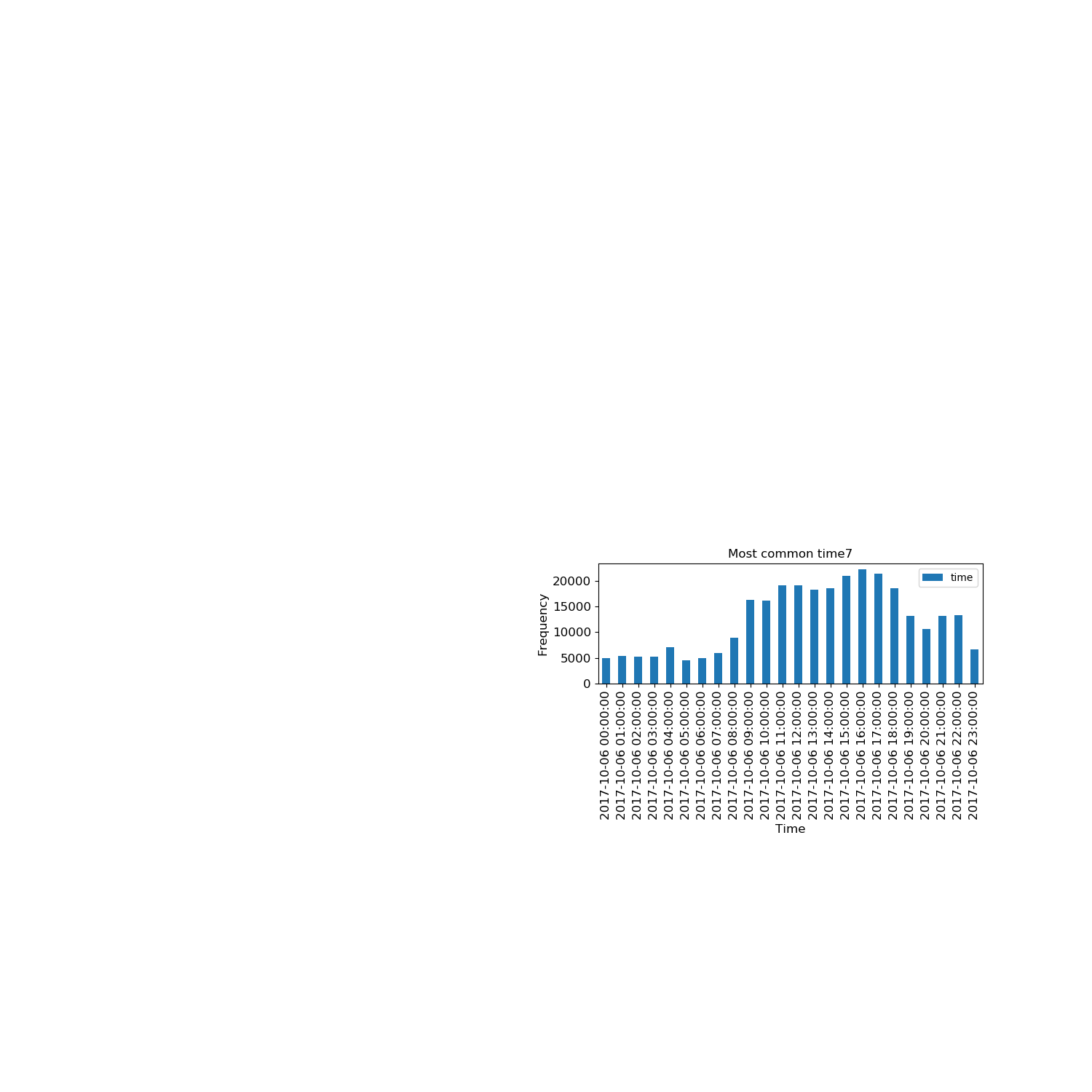
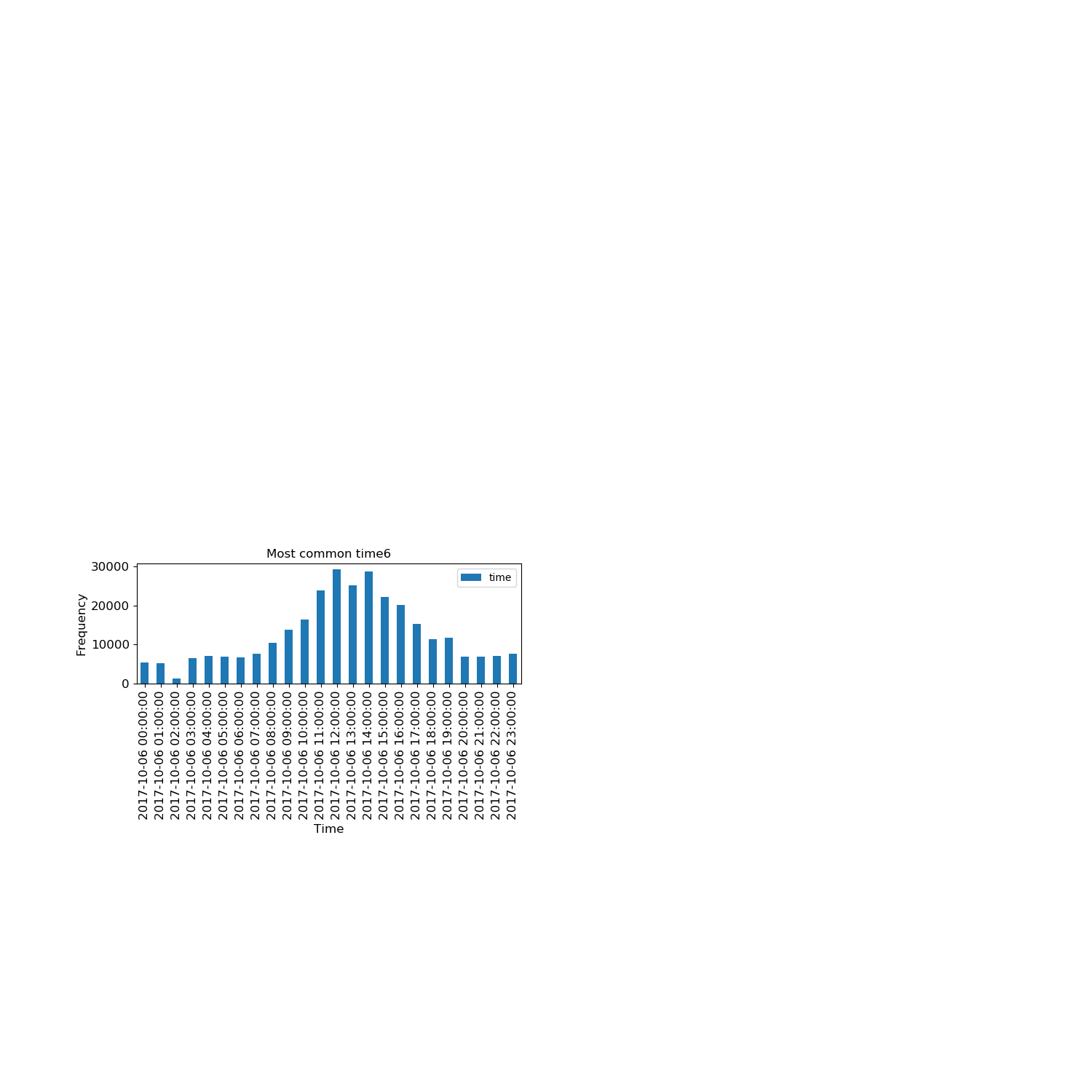
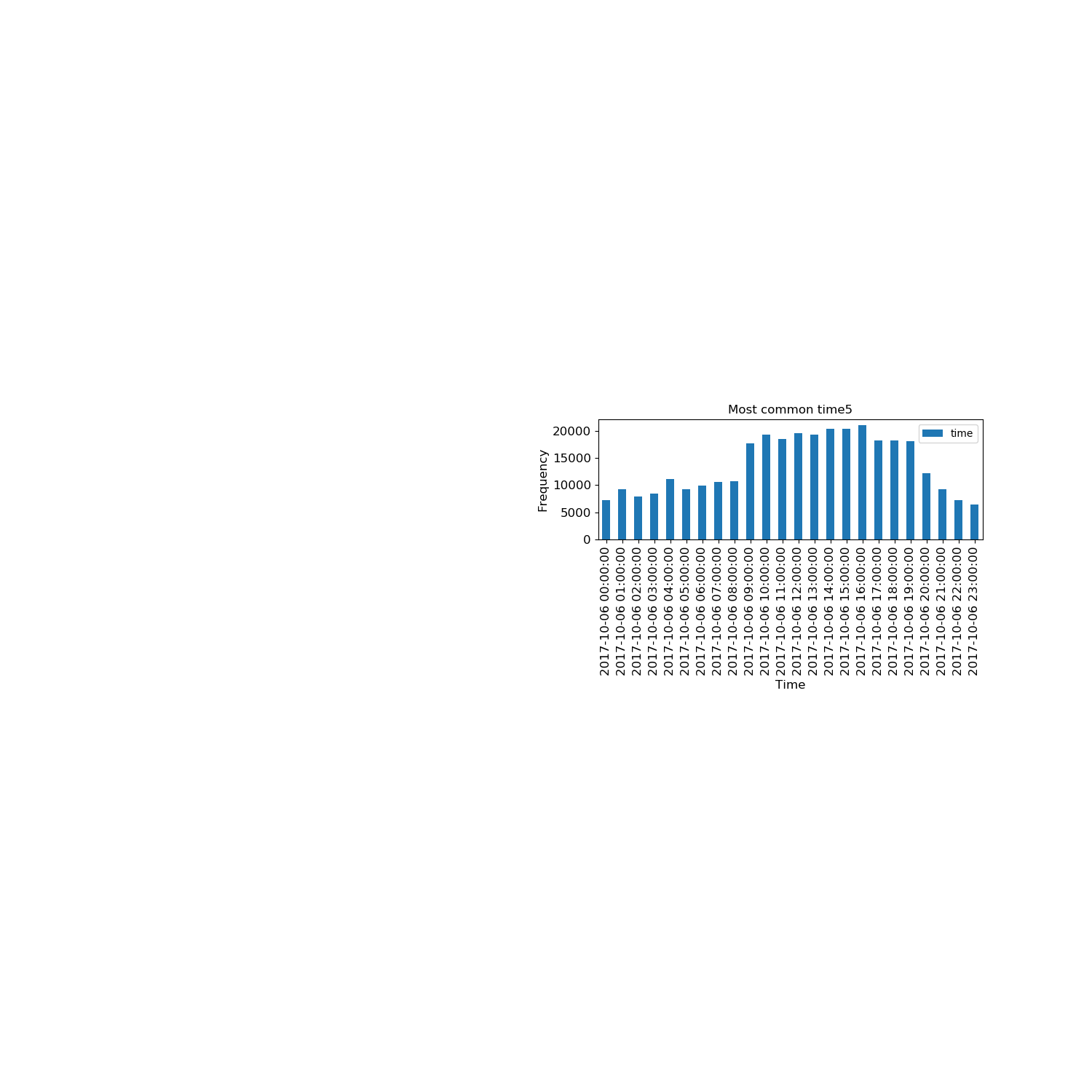
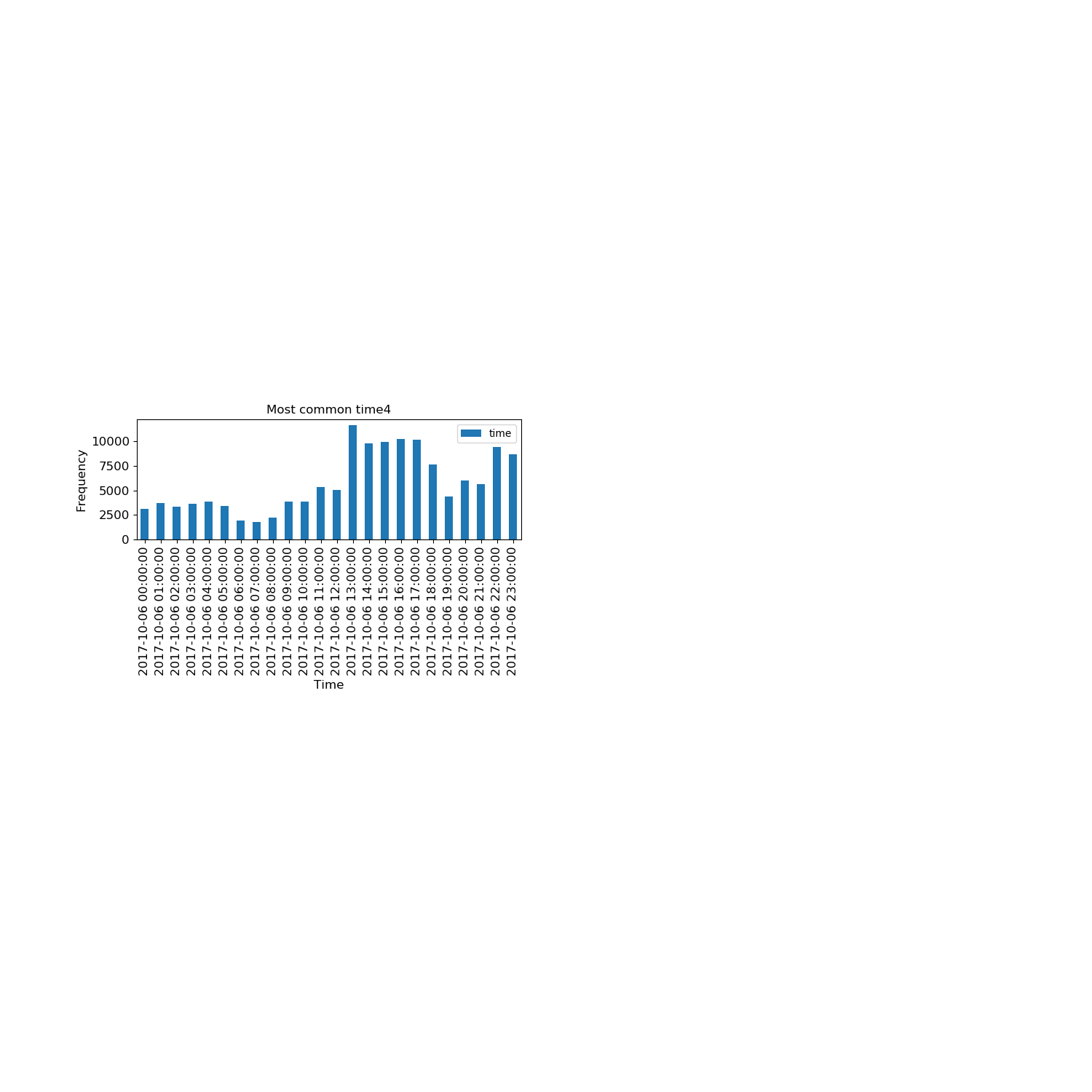
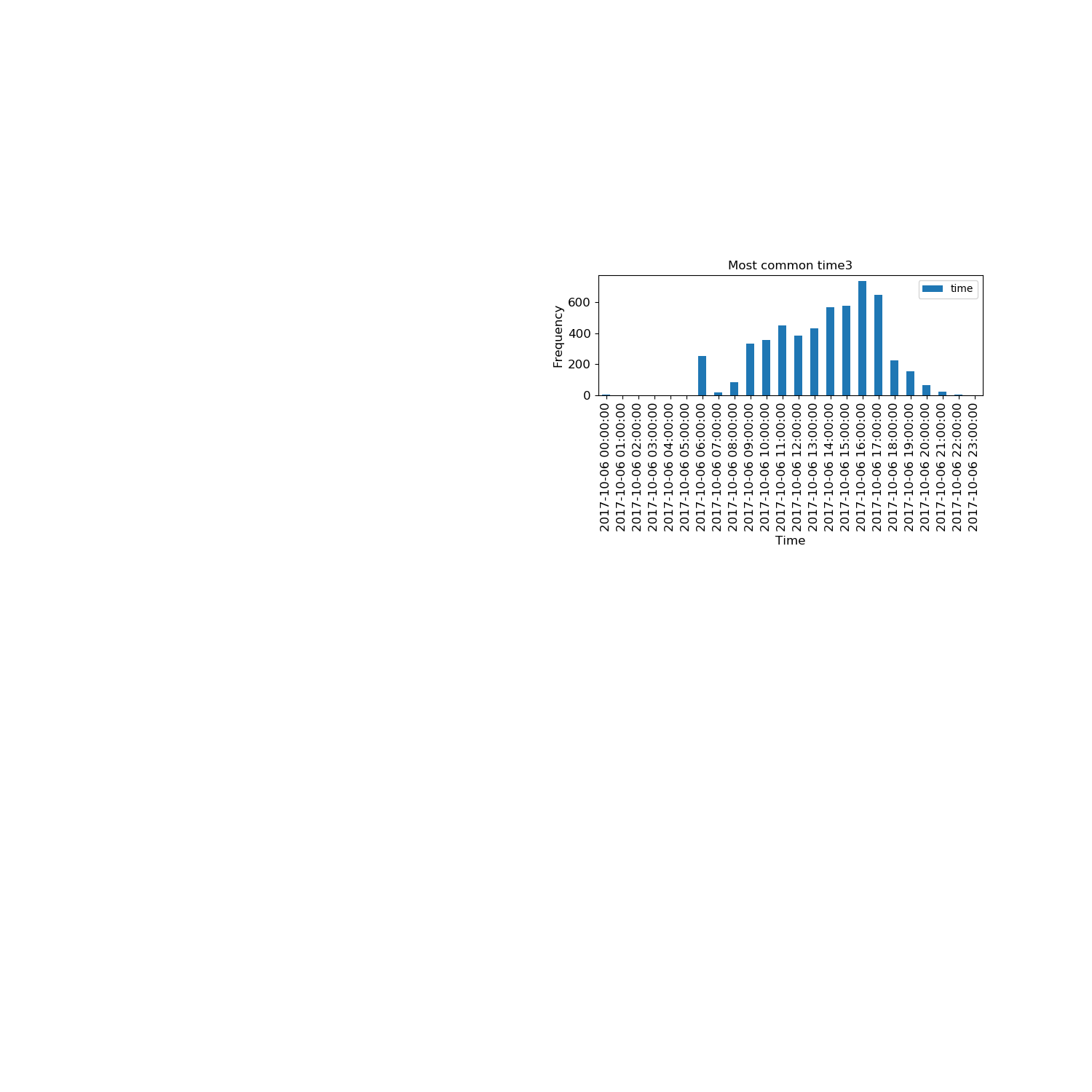
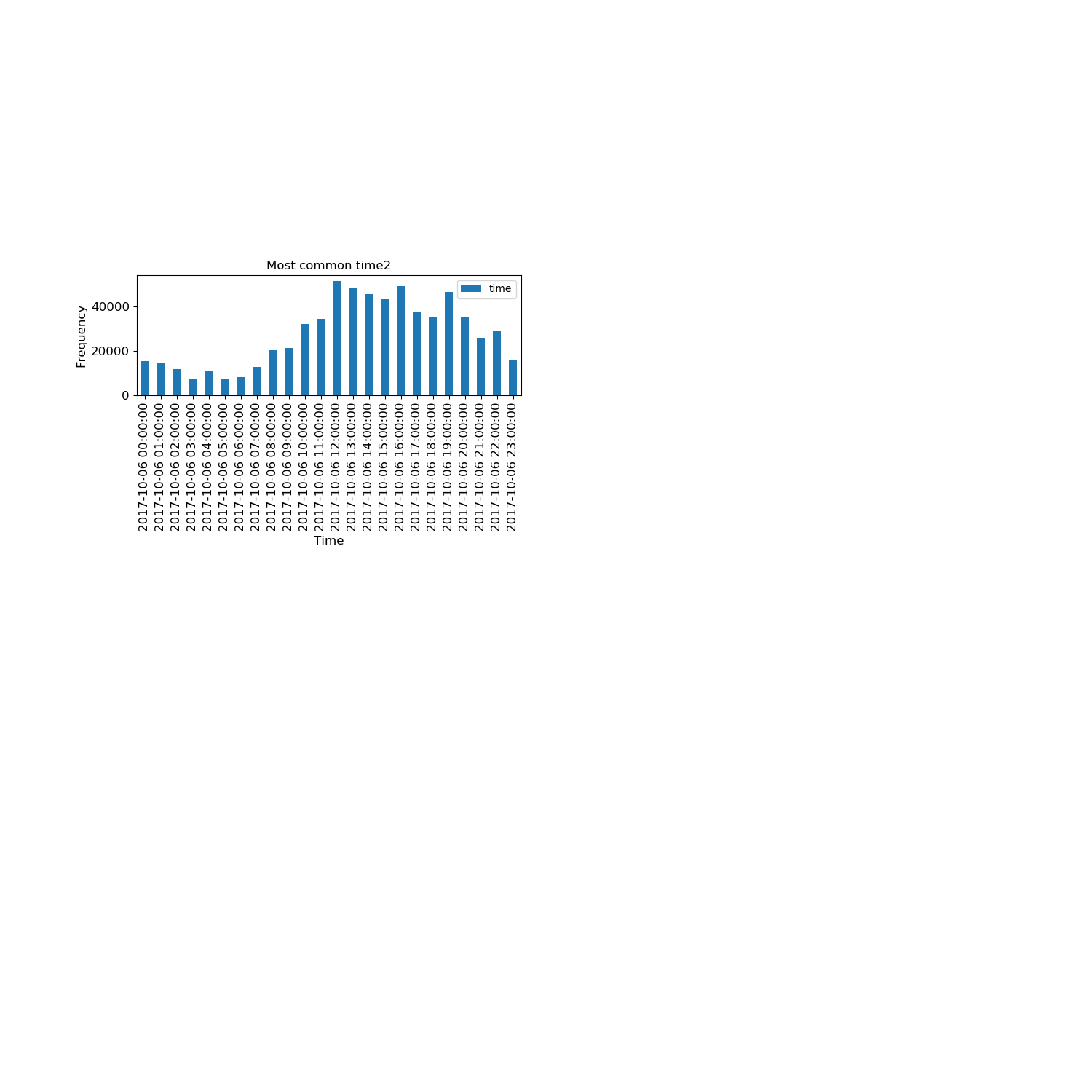
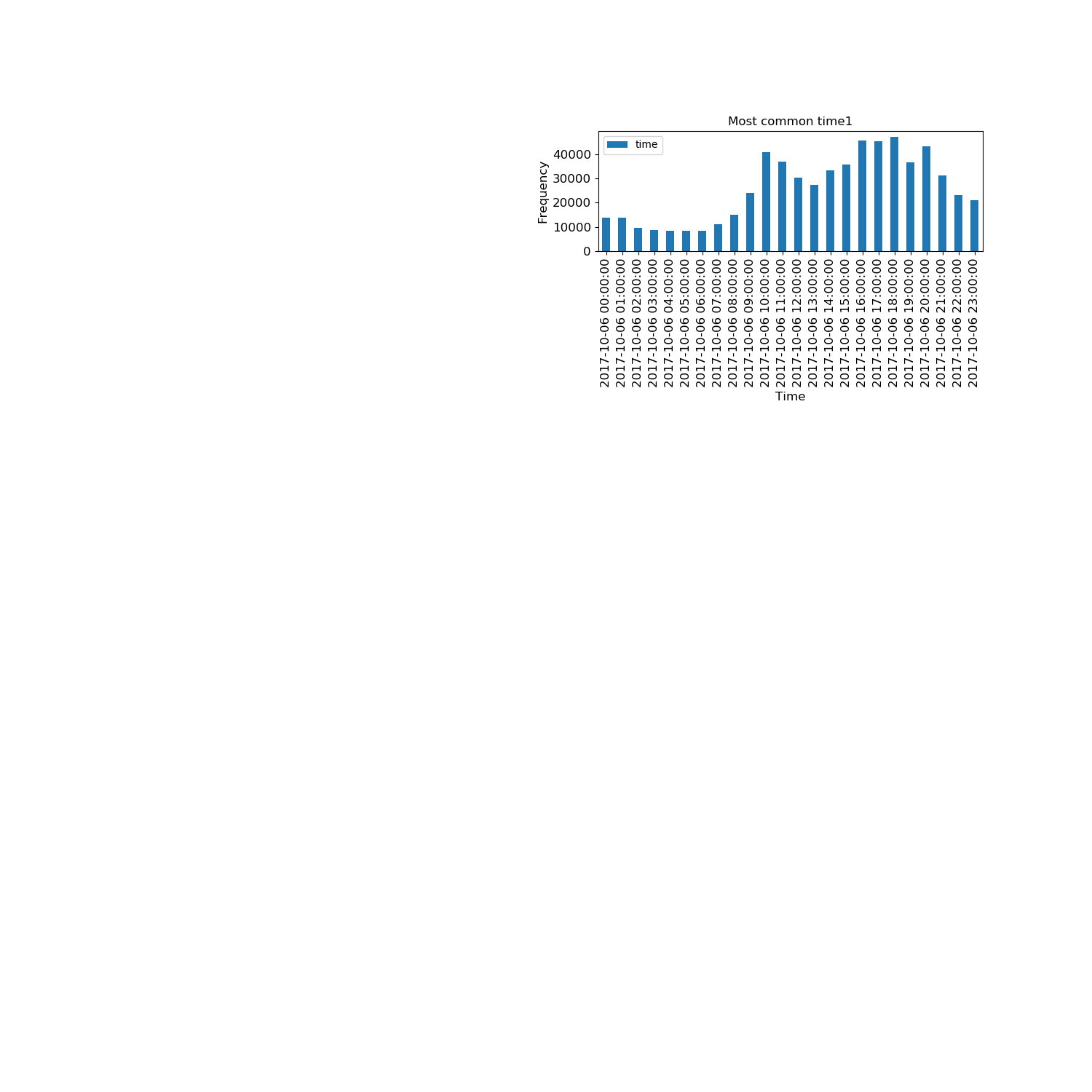
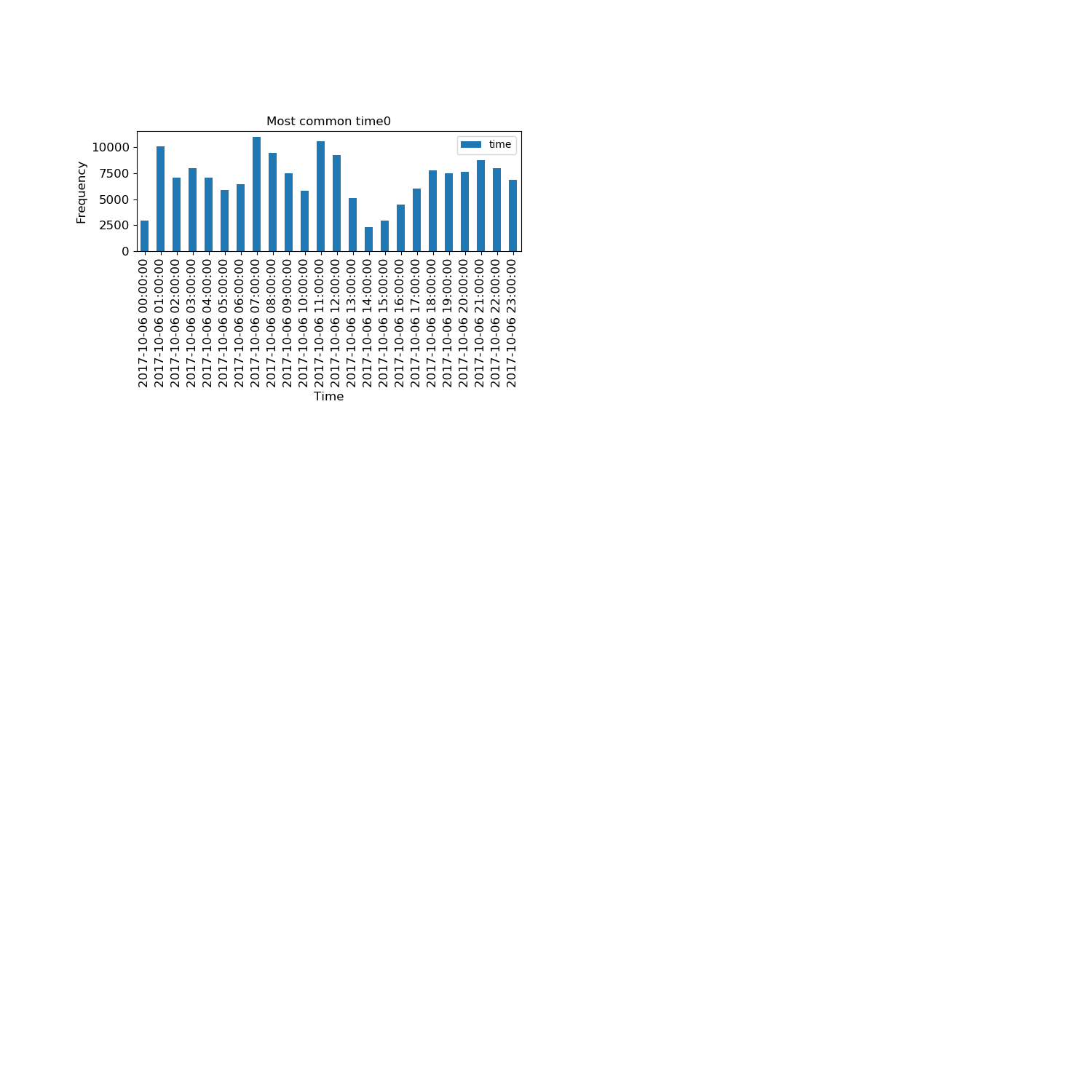
Similarly, the reporting owner CIK for Jack Halperin was among the top 5 CIKs for two months in 2005. Other recurring popular CIKs: Google, IBM, Alexanders J Corp, and Macromedia Inc which was purchased by Adobe that year.

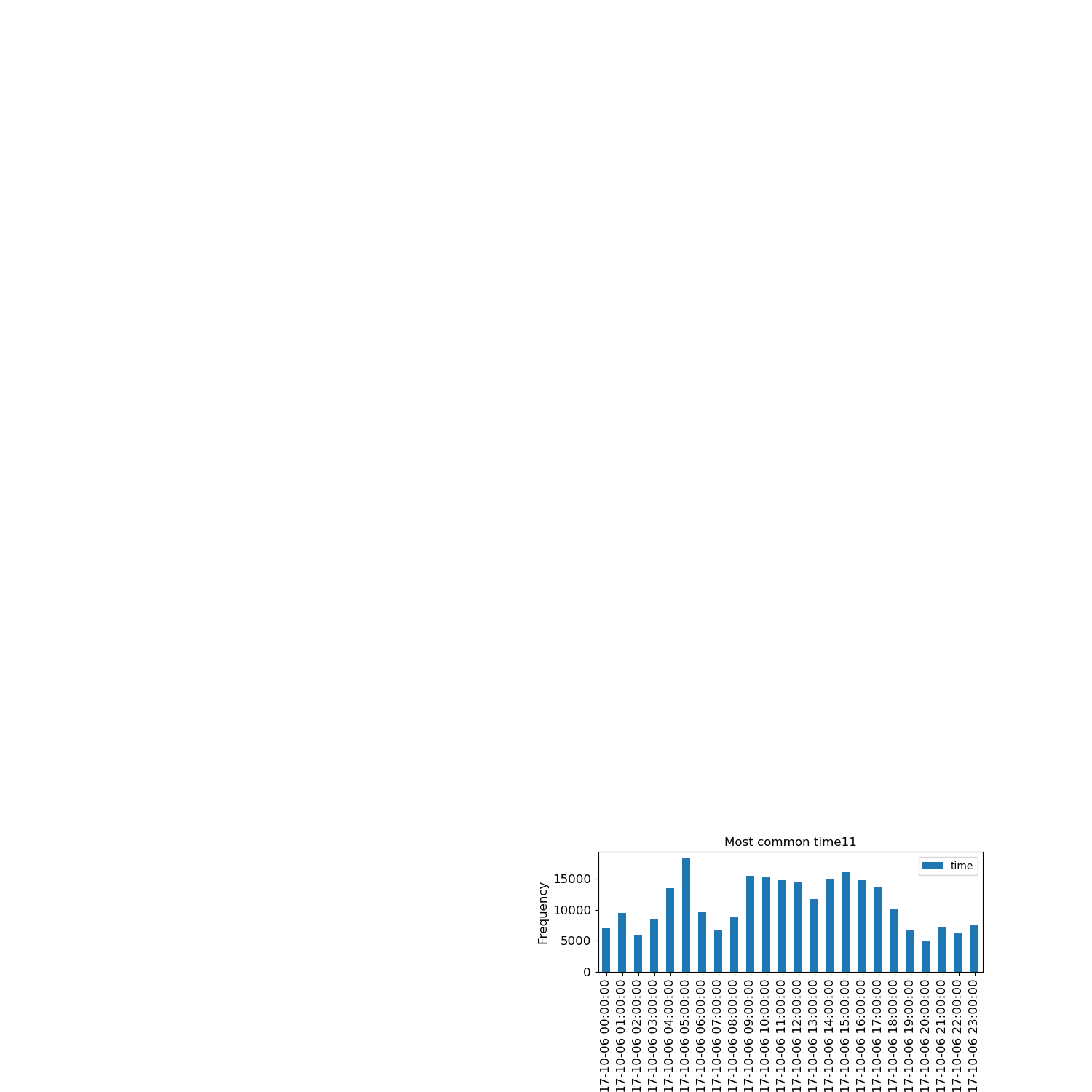
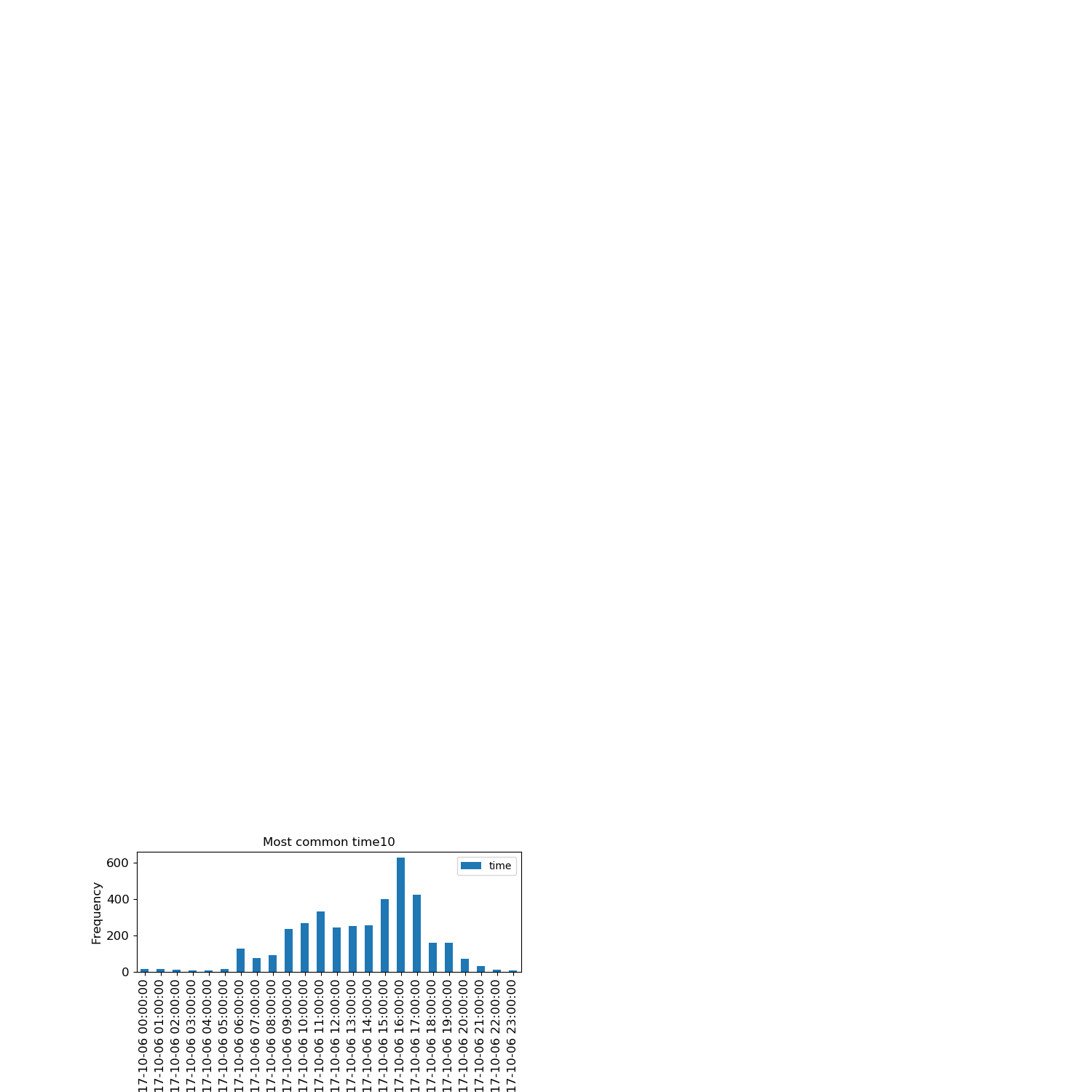
In examining the hourly activity histograms, there is a general pattern of more activity occurring during EST business hours (possibly explained by the concentration of financial districts on the US east coast). The one day that completely breaks from this pattern (10/1/05, "Most common time9") was a Saturday, which would be consistent with the majority of traffic occurring during business hours. On that day the traffic numbers are in the hundreds rather than tens of thousands, with a peak at 1 AM most likely from overnight scripts and crawlers.

The file sizes in the logs contain a large number of "outliers**"** on the scale of tens of gigabytes as identified by the box plots. However, there are so many outliers in many of the logs that replacing them with a different value (e.g. 1.5 standard deviations) would significantly alter the shape of the data and thus would not be appropriate. For example, in the October 2005 data ("Size outliers9") there are two files that significantly skew the data and might be appropriate for removal depending on the purpose of the analysis, however in the February 2005 data ("Size outliers1") there are so many outliers over 20 GB that they cannot be counted simply by looking at the plot. Removing them from the data set would change the overall distribution, and could negatively impact analysis used to determine storage space, average network usage, etc. Furthermore, in looking at the box plots in combination with the hourly frequencies, the days with discernible outliers are the days with lower traffic. We thus might assume that theses outliers occur proportionate to the total traffic.

**Frequencies of Edgar data access by hour:**

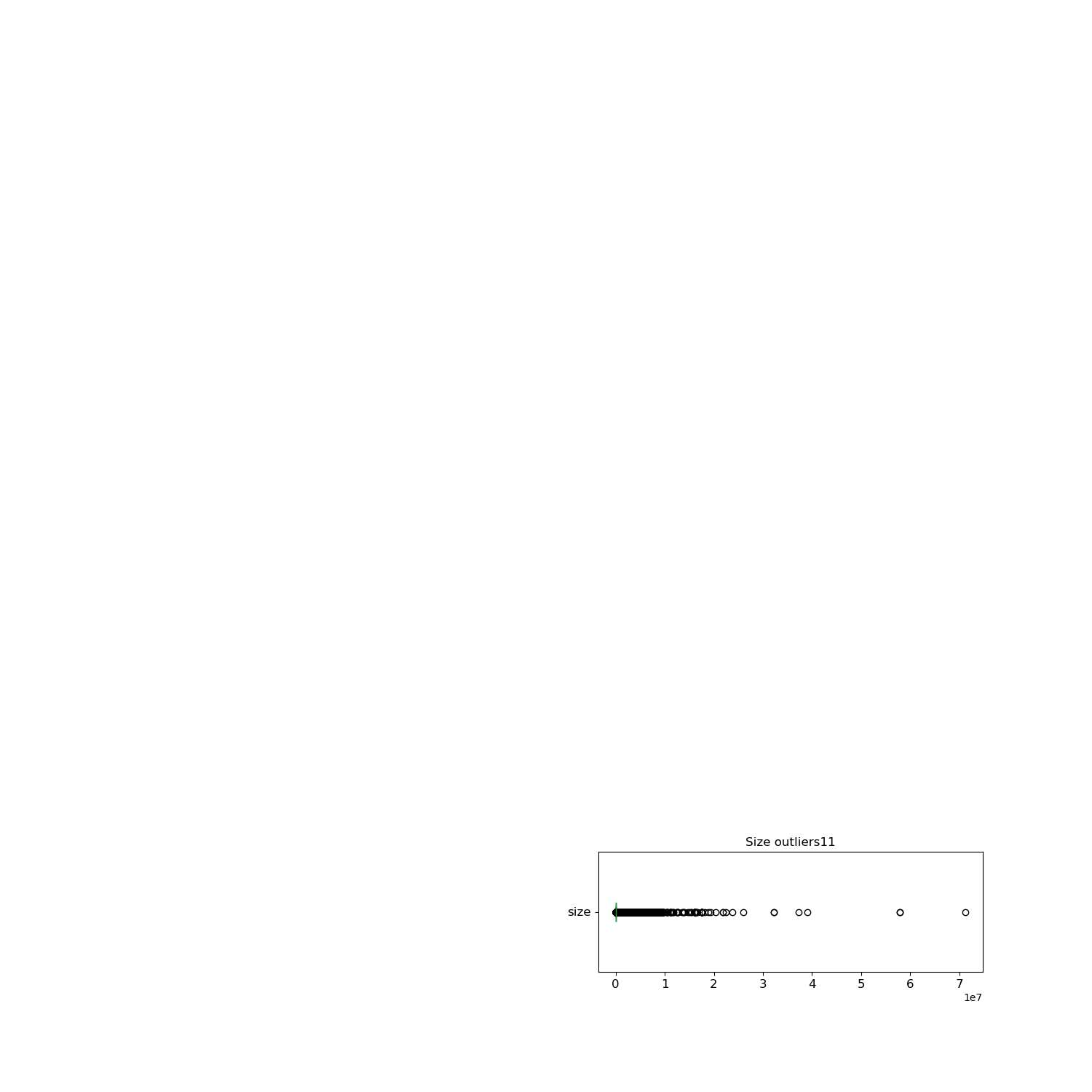
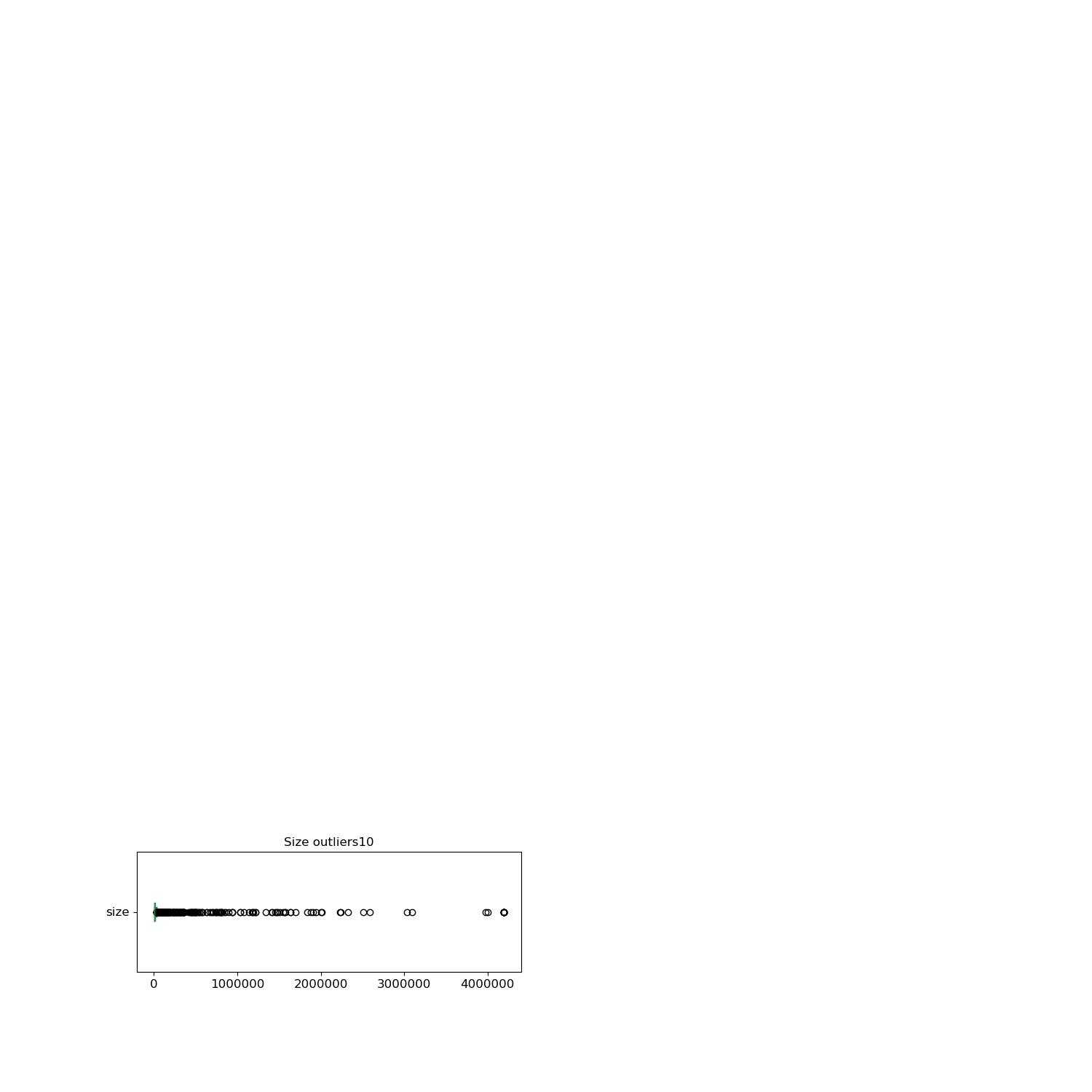
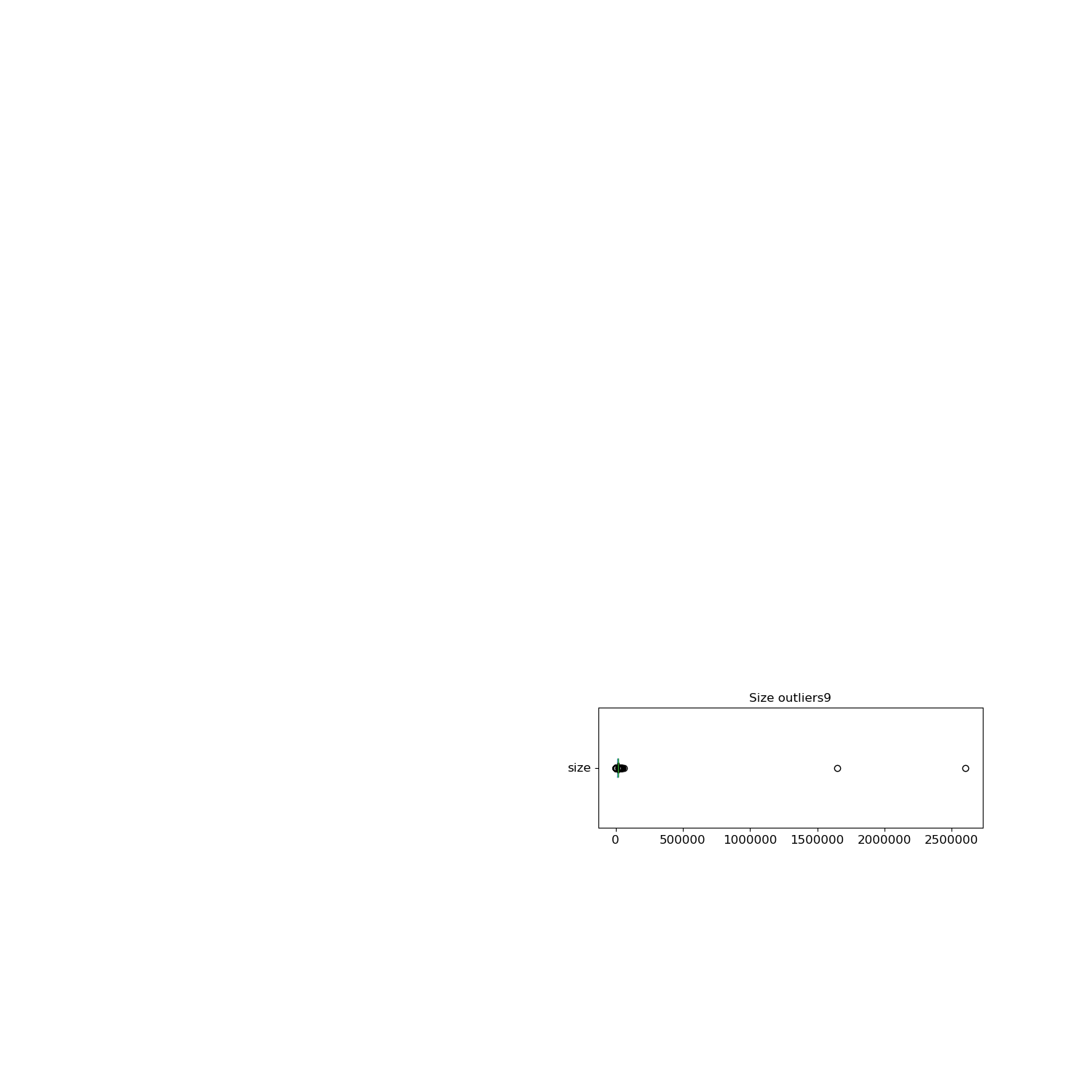
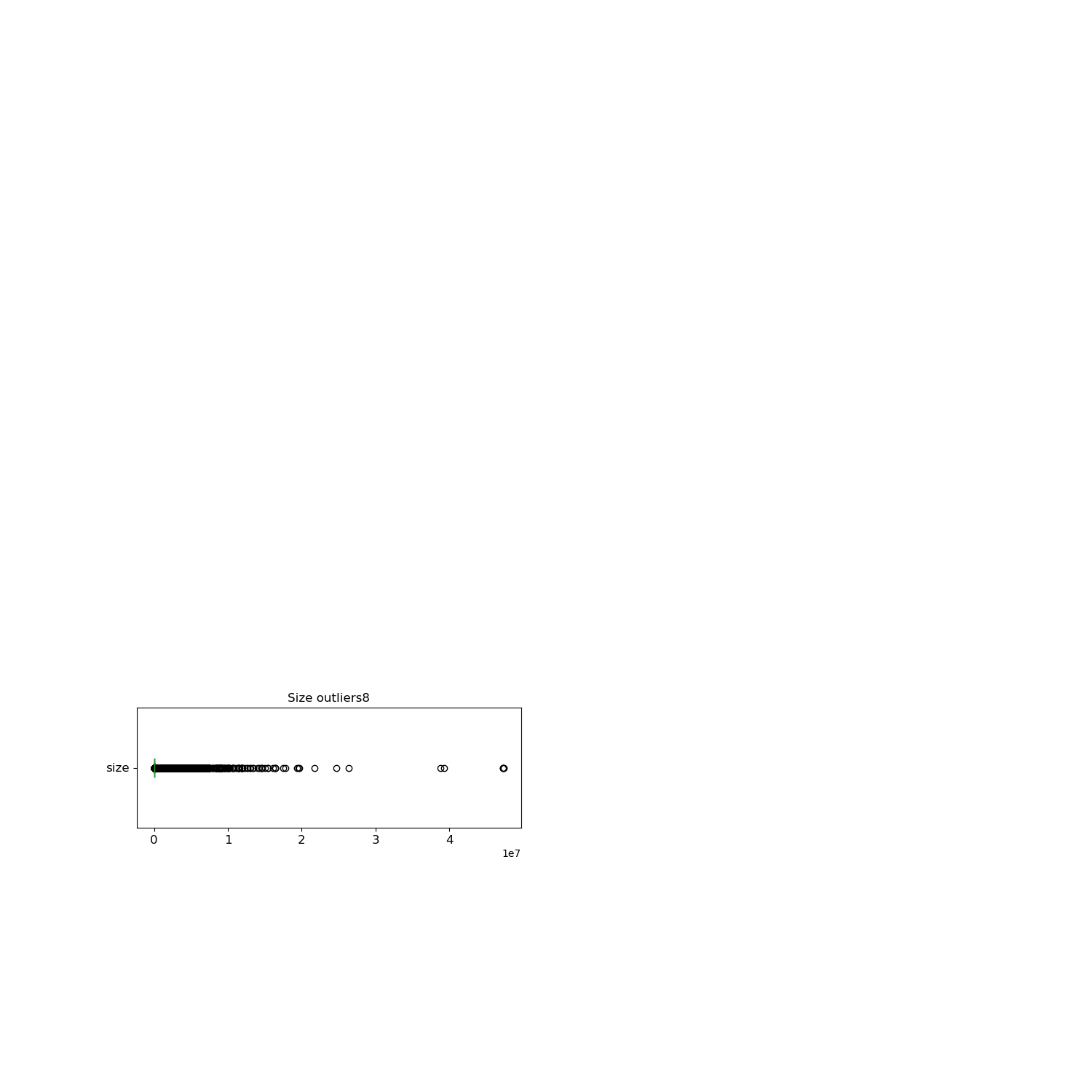
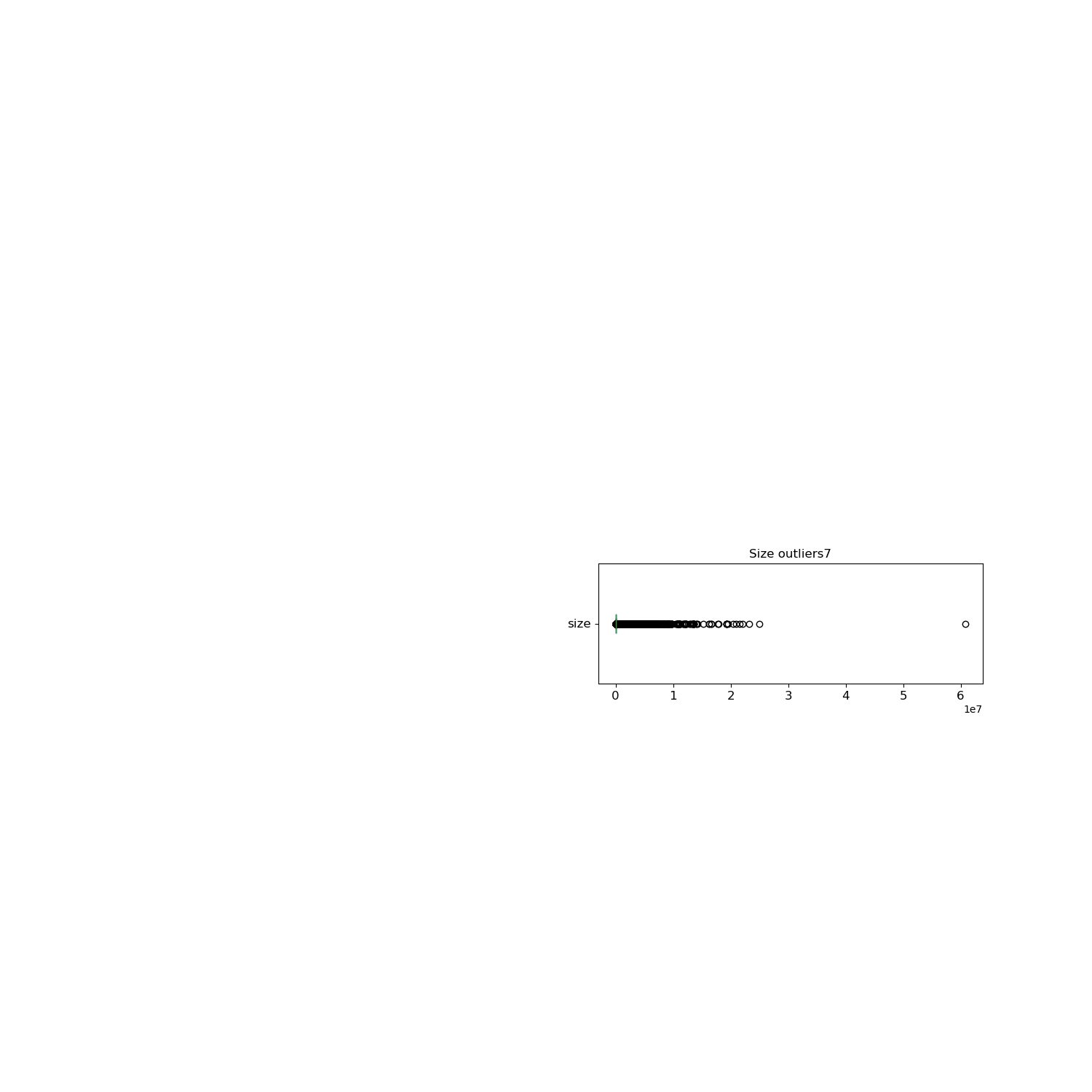
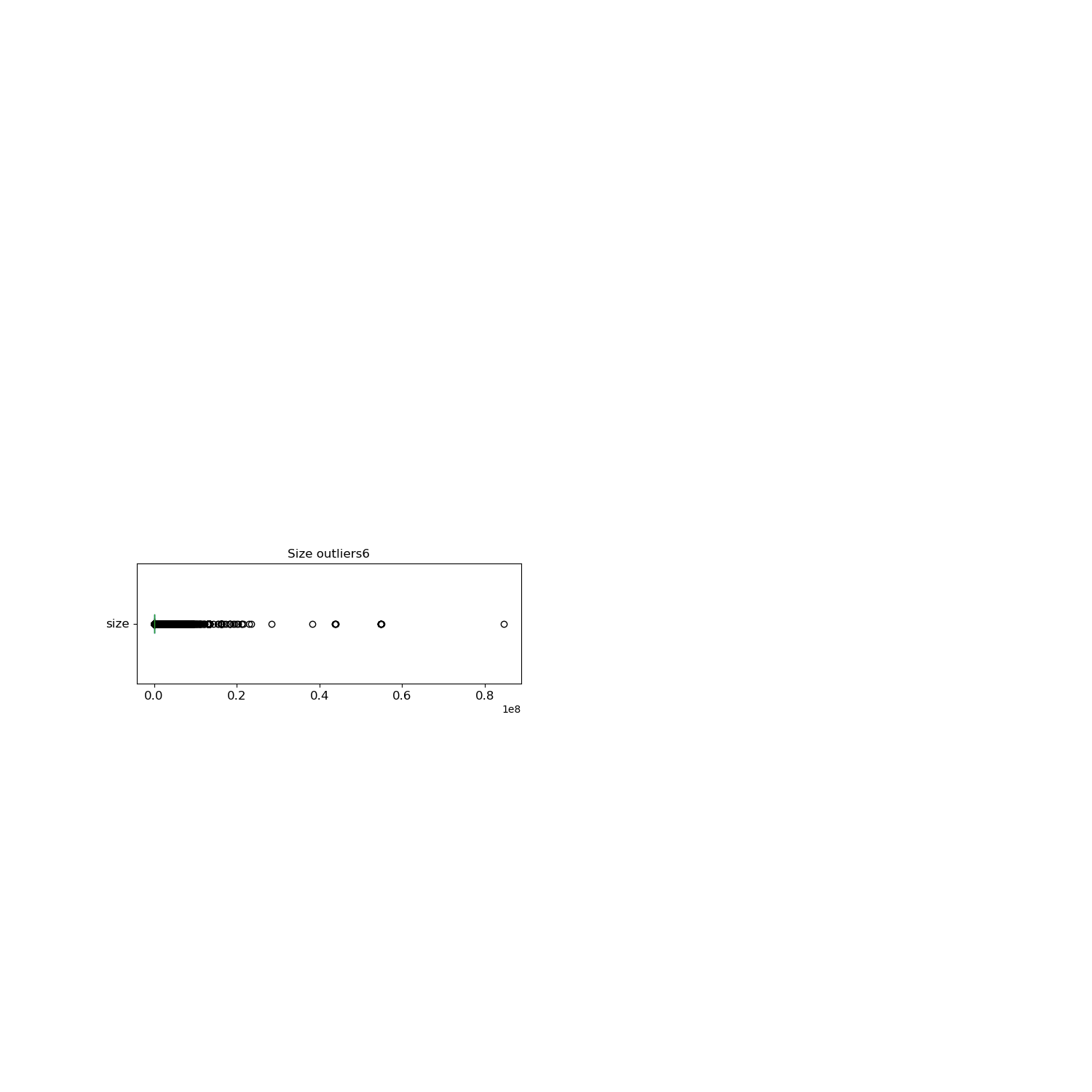
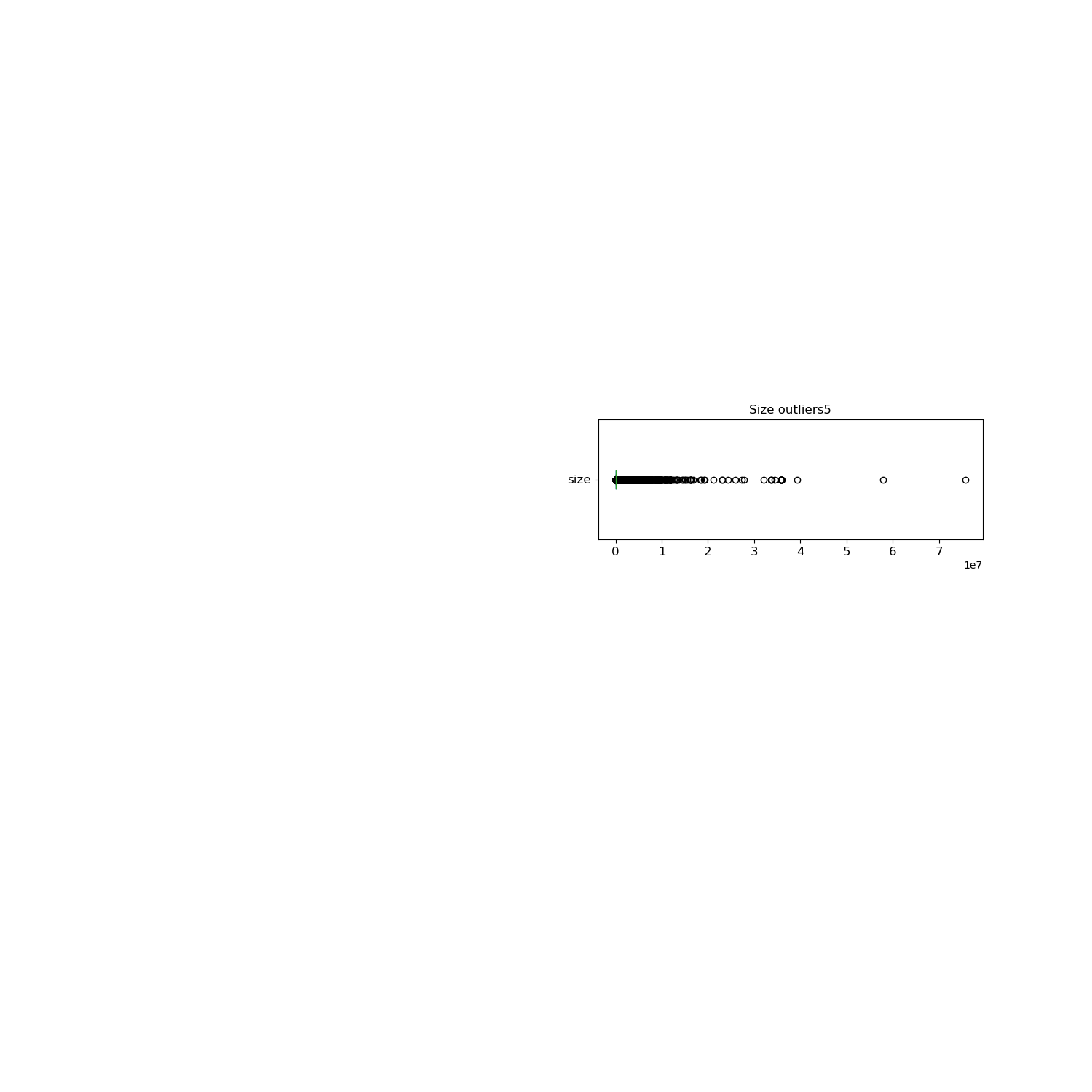
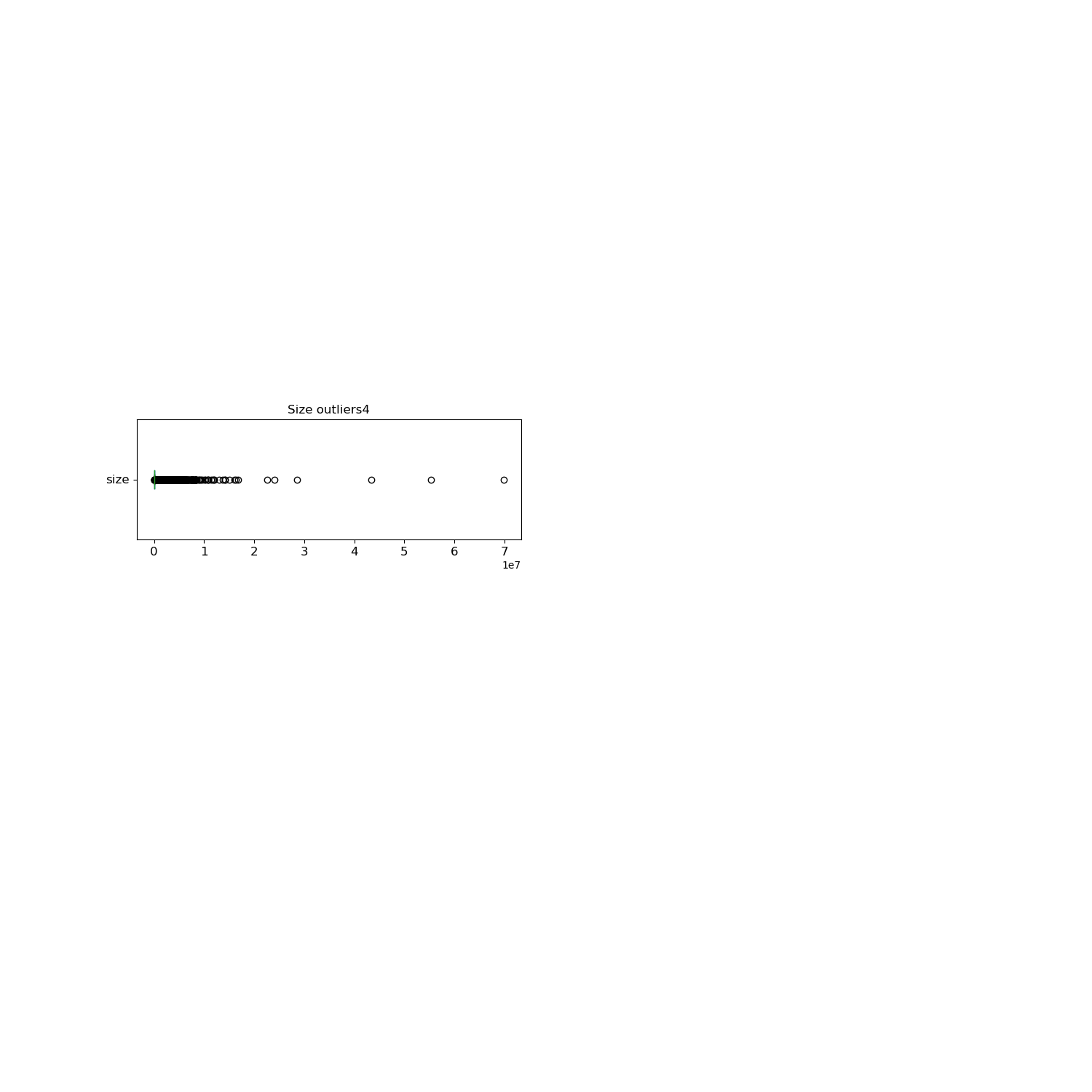
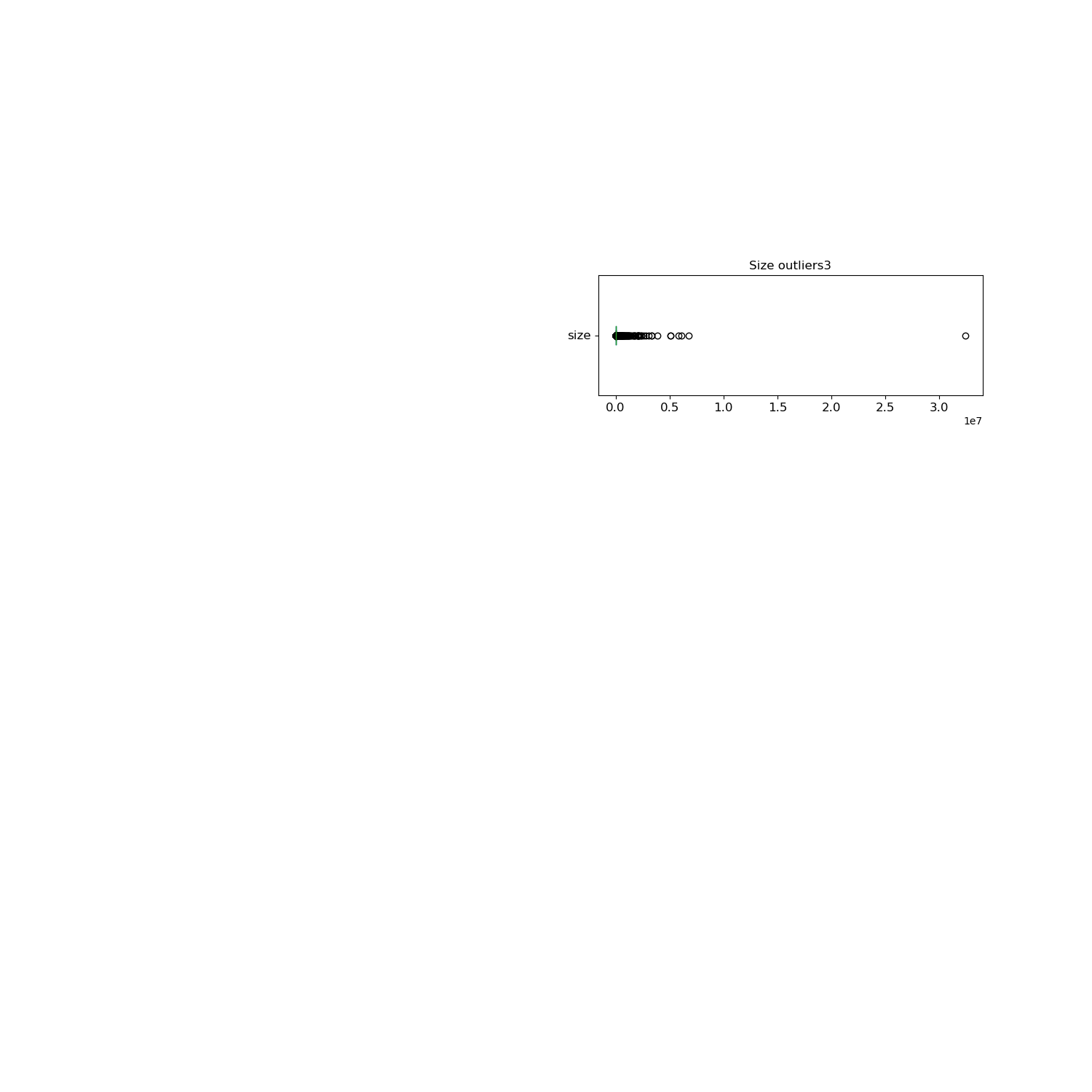
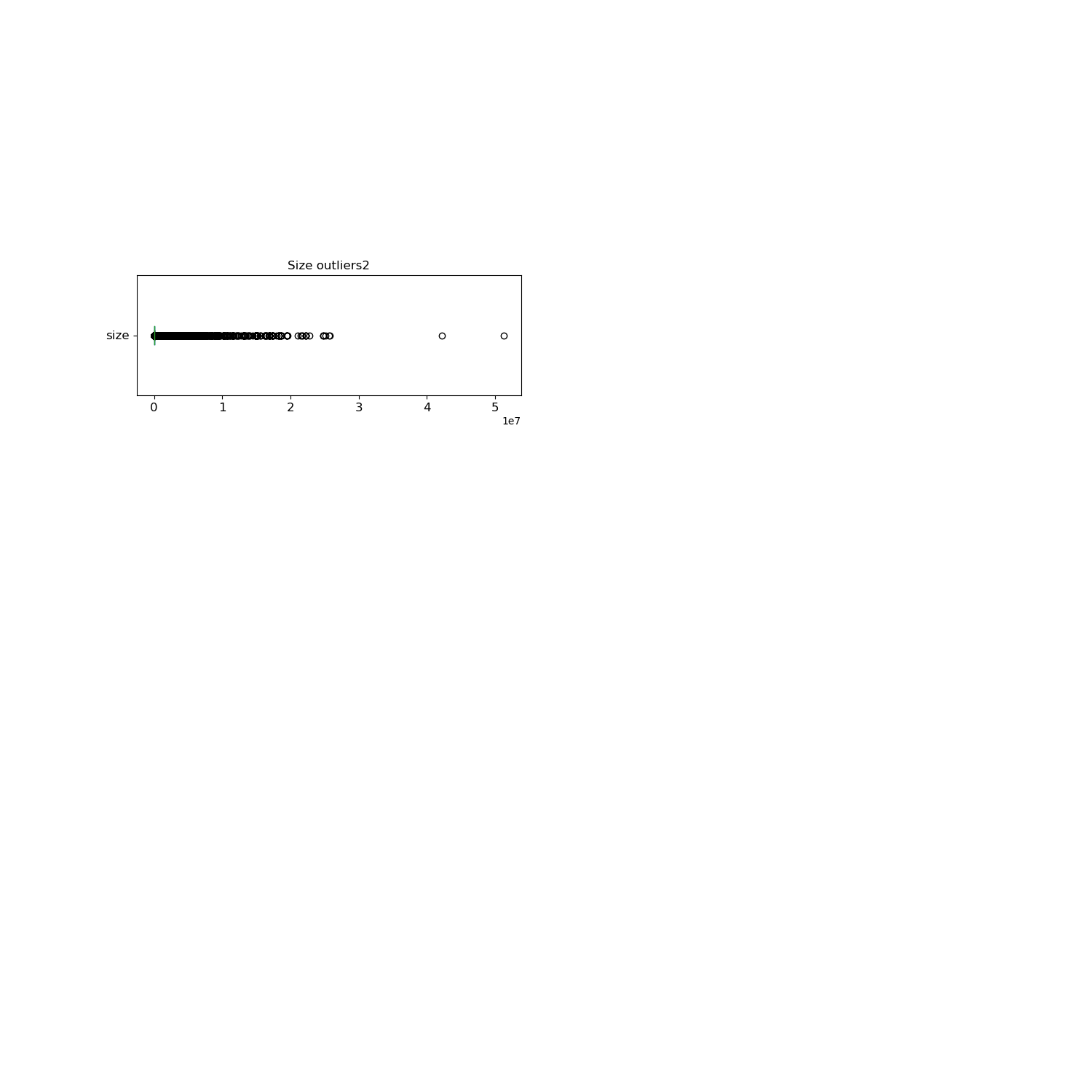
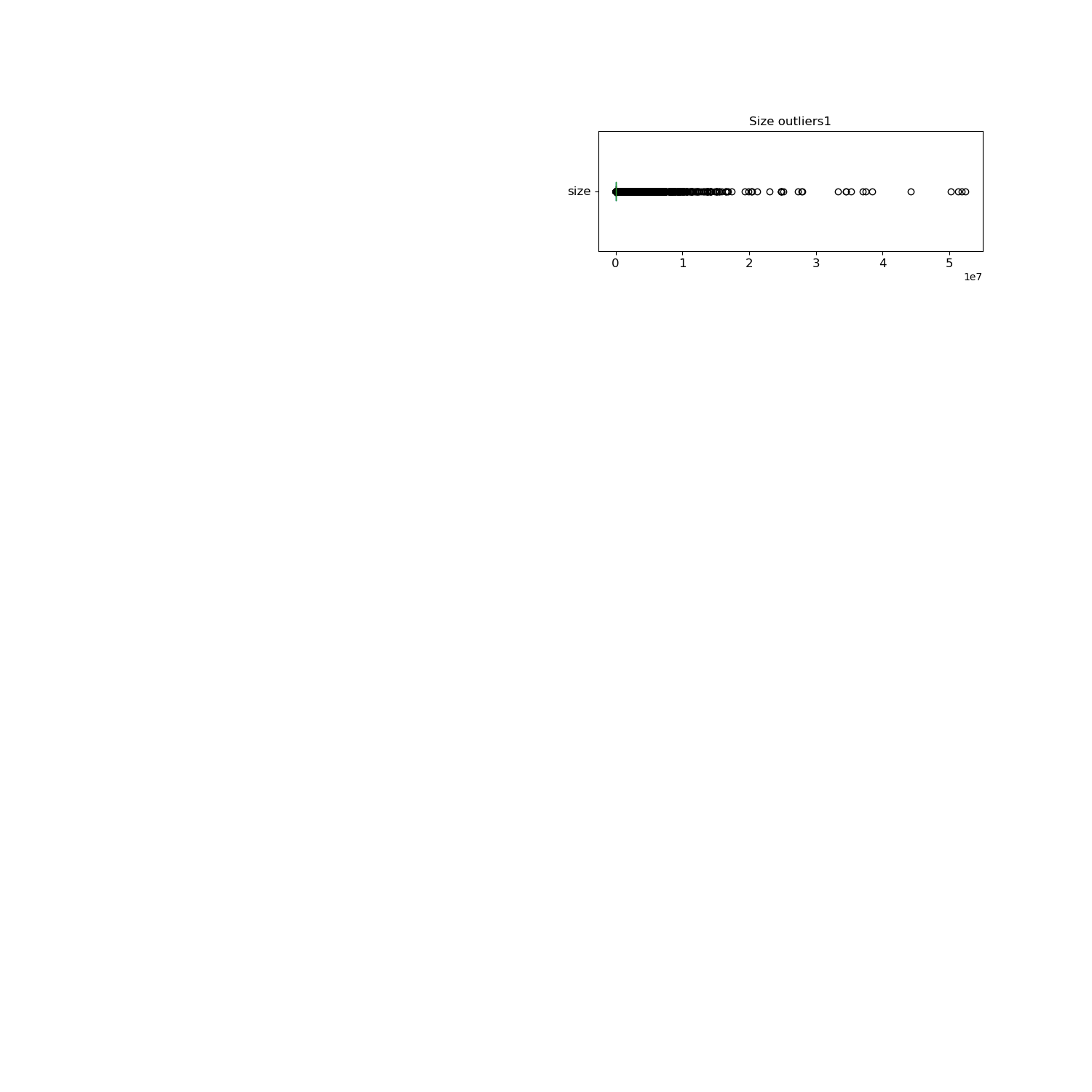
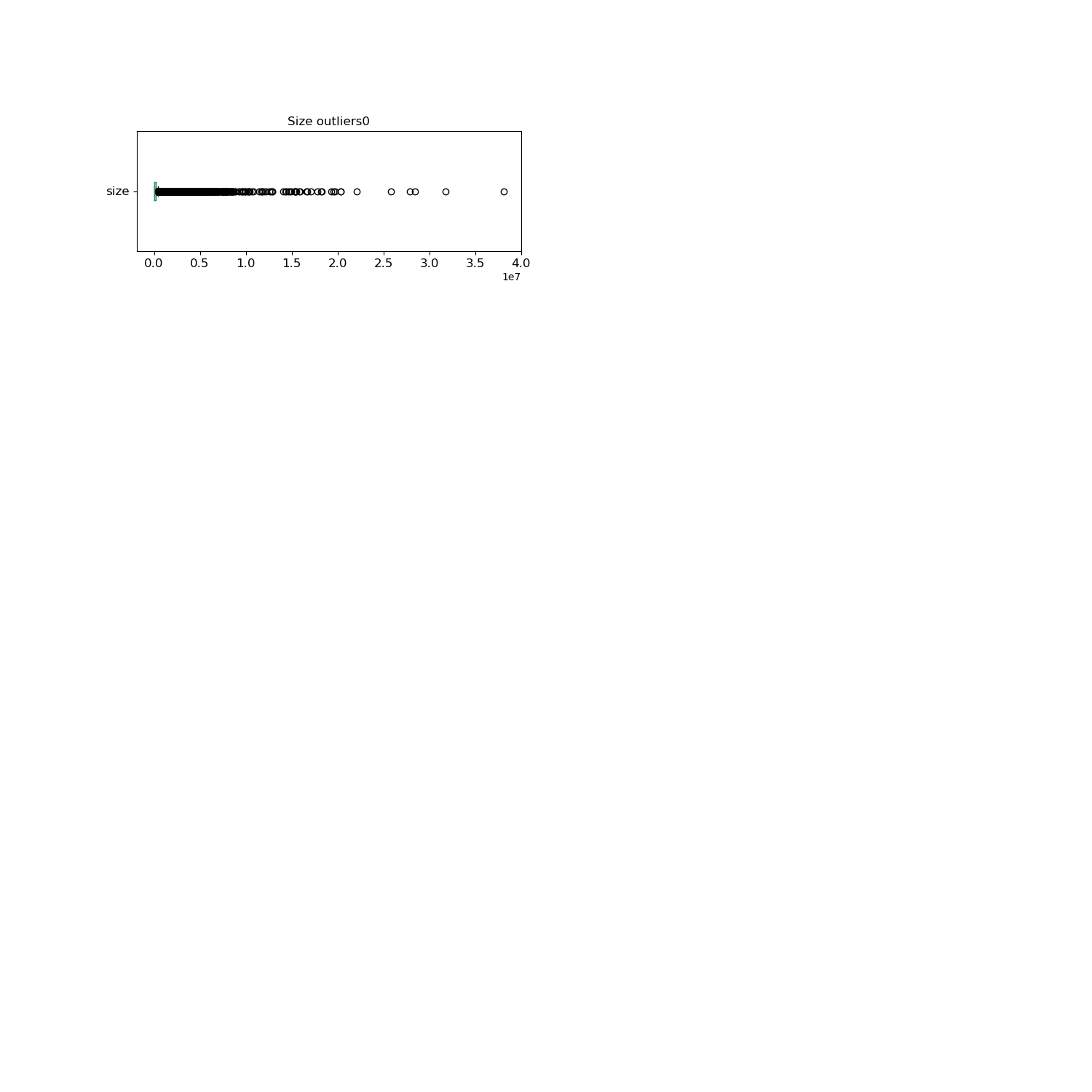
Notes: Month 0 = January, Month 1 = February, etc. Due to an oversight, only times were used for the datetime indices so the date defaulted to the date the code was run.

****

****

**File Size Boxplots:**

Notes: Month 0 = January, Month 1 = February, etc

****