TALLINN UNIVERSITY OF TECHNOLOGY

School of Information Technologies

Maria Kapitonova 220675IABB

Visualizing product and engineering metrics in Jupyter notebooks at Pipedrive

Bachelor's thesis

Supervisor: Inna Švartsman

Tallinn 2024

TALLINNA TEHNIKAÜLIKOOL

Infotehnoloogia teaduskond

Maria Kapitonova 220675IABB

Pipedrive'i toote- ja insenerinäitajate visualiseerimine Jupyter-notebooks

Bakalaureusetöö

Juhendaja: Inna Švartsman

Tallinn 2024

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Maria Kapitonova

18.05.2024

Abstract

The following thesis will describe how Jupyter Notebooks were applied for the visualization of product and engineering metrics at Pipedrive, describing their movement from traditional instruments like Zeppelin to way more dynamic environments. It maintains the idea that Jupyter Notebooks enhance data gathering and subsequent analysis and visualization processes to create actionable insights into Pipedrive's product and engineering operations. The result was an all-encompassing guideline developed to Pipedrive employees' specs: metrics, visualization techniques, and best practices. This yielded a set of metrics, among which the P_99 latency metric has aided optimization and improved system performance and user experience. The thesis further details a pilot implementation of this set of metrics in one of Pipedrive's business units to estimate the impacts on operational efficiency and decision-making. Some feedback from the team has also been taken further to propose some future enhancements, such as including time series analysis.

This thesis is written in English and comprises 34 pages, including 6 chapters, 9 figures, and 1 table.

Annotatsioon

Järgnevas lõputöös kirjeldatakse, kuidas Jupyter Notebooks'i rakendati Pipedrive'is toote- ja insenerinäitajate visualiseerimiseks, kirjeldades nende liikumist traditsioonilistelt vahenditelt nagu Zeppelin palju dünaamilisematesse keskkondadesse. See säilitab idee, et Jupyter Notebooks parandab andmete kogumist ja sellele järgnevaid analüüsi- ja visualiseerimisprotsesse, et luua rakendatavaid teadmisi Pipedrive'i toote- ja inseneritegevusest. Tulemuseks oli Pipedrive'i töötajate spetsifikatsioonide järgi välja töötatud kõikehõlmav juhend: mõõdikud, visualiseerimistehnikad ja parimad tavad. Selle tulemusel saadi hulk mõõdikuid, mille hulgas on P_99 latentsuse mõõdik aidanud optimeerimist ning parandanud süsteemi jõudlust ja kasutajakogemust. Bakalurusetöös kirjeldatakse lisaks selle meetrikakomplekti pilootrakendust ühes Pipedrive'i äriüksuses, et hinnata selle mõju tegevuse tõhususele ja otsuste tegemisele. Mõnda meeskonnalt saadud tagasisidet on võetud ka edasi, et teha ettepanekuid mõnede tulevaste täiustuste kohta, näiteks lisada aegridade analüüs.

Käesolev lõputöö on kirjutatud inglise keeles ja koosneb 34 leheküljest, sealhulgas 6 peatükist, 9 joonisest ja 1 tabelist.

List of abbreviations and terms

API	Application programming interface
RAM	Random access memory
IDE	Integrated Development Environment

Table of contents

List of figures

List of tables

1 Introduction

Data visualization is an essential function of business intelligence tools and the key to advanced analytics. Visualization helps in assessing the value of information or data created today. Data visualization refers to presenting information in a graphical form, such as a pie chart, graph, or other types of visual representation. [11]

Pipedrive is a firm that offers software for managing customer relationships. [12] To enhance the collection, analysis, and visualization of product and engineering metrics, the company needs to transition to Jupyter Notebooks from Zeppelin. This transition involves not only a change in tools but also the need for a deep understanding of the capabilities of Jupyter Notebooks to ensure that all departments across the company are working effectively together.

1.1 Background and Problem

Pipedrive uses many smart tools to make its work faster and better. But old tools don't always bend or shift fast to meet new market needs. Jupyter Notebooks can fix these issues with their bendy nature and strong data study powers. Yet, making Jupyter Notebooks fit into Pipedrive's tech setup and my team brings some hard tasks. This means gathering, handling, and showing data right for what each part of the company needs. It's also key to create ways to teach and help users improve their use of this fresh tool.

1.2 Objectives of the work

The goal of this work is to build a simple system with Jupyter Notebooks to make metrics for looking into product and engineering info at Pipedrive. This will improve the data-gathering, study, and visualization steps, causing clearer insights into important parts of the firm's product and engineering efforts.

Tasks to be performed:

- Data Analysis and Visualization: Use Jupyter Notebooks to collect, process, and visualize data gathered from the metrics. Develop user-friendly dashboards and reports that help employees comprehend the current state of projects and identify potential issues.

- Guide: Offer a complete manual on how to make and use metrics in Jupyter Notebooks to look into product and engineering information for my team at Pipedrive.
- Putting into action and Testing: Conduct a pilot implementation of the developed metrics and analytical tools in one of Pipedrive's business units. Analyze their impact on work processes and evaluate the outcomes.

2 Background and theoretical foundations

2.1 Jupyter Notebooks

Jupyter Notebook is a notebook program for writing, transferring, and running code. It can be used as a development environment. It exists as a web service, i.e., it is accessible via the Internet and allows you to transfer code to other developers. [6] This simplifies the data analysis process and makes it more visual and reproducible.

2.1.1 Broad-Spectrum Languages and Dynamic Data Exploration

The environment is often used for Python but also exists for other programming languages. Jupyter Notebook supports Ruby, Perl, R, MATLAB, Julia, and other languages. [7] These are often specialized languages for tasks involving quickly writing and executing a small program.

Creating analytical reports, articles, and interactive paragraphs for textbooks is also possible. That's why Jupyter Notebook is used in data analytics and statistics, where you often need accurate reports and visualizations with results. [8]

2.1.2 Challenges in Reproducibility and Usage Discipline

Researching over 1.4 million Jupyter notebooks stored on GitHub revealed that although there are notable advantages, many notebooks face reproducibility challenges. Just 24.11% of the notebooks ran error-free, with only 4.03% producing the same results on a second run, showing issues with dependency management, hidden state, and non-linear execution order greatly impacting study reproducibility. [1]

2.2 Zeppelin notebooks

Zeppelin is an open-source tool in a web notebook format designed for multi-purpose workspaces dedicated to data analysis. This multifunctional interactive shell allows you to run queries to various data sources and process and visualize the results. [3] Zeppelin, like Jupyter, looks to the user like a set of notebook files consisting of paragraphs in which queries are written and executed. With the help of built-in visualizers, a laptop with a set of queries can be easily turned into a full-fledged dashboard with data. [4]

2.2.1 Functionality

Zeppelin is a web interface for various databases and can act as an interactive shell for executing scripts in programming languages. It includes interpreters for R and Python, so it may act as an alternative to the usual RStudio and Jupyter. [3]

Zeppelin can obtain data via API from third-party services and has the ability to process data in addition to regular queries to the database and automate these processes. Zeppelin supports updating dashboards by crown without unnecessary movements. It also has a built-in version control system. It has several language interfaces and Apache Spark integration, which allows you to solve various analytical problems inside notebooks. [3]

2.2.2 Challenges

Zeppelin has its drawbacks: it still does not always work reliably (it is open source), the web interface eats up a lot of RAM, and some may lack the functions of a full-fledged IDE. [5]

2.3 Jupyter notebooks vs Zeppelin notebooks

Both tools are open-source and are web-based notebooks for data development and visualization.[9] However, Jupyter is positioned as a multilingual interactive computing environment with support for code, equations, text, graphs, and interactive dashboards.

Apache Zeppelin does not aim to take the place of an IDE. However, it does include some software development features, focusing on capabilities for interactive big data analysis:

1. Wide range of language assistance. Jupyter is more versatile for data analysis because it can be used with a variety of programming languages. [10]

2. Seamless integration and superior security: Jupyter streamlines the combination method with numerous systems and cloud offerings, imparting robust security features to protect facts and evaluation techniques. [2]

3. Dynamic network and guide. Jupiter's energy lies in its active community. The platform is based on the collective knowledge of a more energetic person and

developer base, allowing short problem resolution and rapid implementation of revolutionary features.

4. Optimized performance and scalability. Jupyter guarantees easy overall performance even if running with large datasets. [9]

The transition to Jupyter is a strategic step that has been taken to simplify the use of gear and expand the talents of information analysis.

3 Challenges in Visualising Metrics

After reading the official documentation of Jupyter Notebooks, as well as various forums and articles written by regular users (such as Medium, Stack Overflow, and Data Science Central), I have highlighted issues and best practices to keep in mind when using and writing notebooks in Jupyter Notebooks.

3.1 Data Integration Complexity

Working with different data sources in Jupyter Notebooks can be complex and affect data analysis's accuracy and validity. These difficulties can arise from several issues:

- Different data formats
- Synchronizing data updates
- Ensuring data consistency

3.1.1 Different data formats

Supporting several programming languages and the ability to interact with various data formats makes Jupyter Notepads flexible. However, integrating data sources with different formats such as CSV, JSON, XML, or databases can be challenging. Each format may require specific handling, parsing, and pre-processing to ensure the data is suitable for analysis.

3.1.2 Synchronization of Data Updates

Maintaining synchronization between the data in Jupyter notebooks and updating it in real-time is challenging. The challenge is not only technical; it ensures that the data is consistent and correct throughout the analysis process.

It is crucial to ensure that each notebook session receives the latest data. To solve this problem, it is usually necessary to establish connections to databases or implement APIs that allow receiving updated data at regular intervals.

3.1.3 Ensuring Data Consistency

When combining data from various sources, maintaining data consistency becomes crucial. It is essential to tackle issues like redundant records, conflicting data entries, and insufficient data. If the notebook contains the results of a study, it should not include anything extra after the work is completed. The notebook should include the minimum number of conclusions that represent the best answer to the question under study. Each assumption and conclusion should always be accompanied by explanatory comments.

Overall, working with different data sources in Jupyter Notebooks requires careful consideration of these issues to maintain data accuracy, reliability, and the validity of data analysis results. [15]

3.2 User Experience (UX) Constraints

The main problem with the solutions listed below is the need to install and configure multiple extensions, which requires additional effort and knowledge from the operations team.

3.2.1 Data visualization

In Jupyter, you can and should use libraries to visualize data in Python: matplotlib, plotly, Seaborn, Bokeh, and others. However, you have to learn a new library to do this and write new code to change the plot type. This does not fit well with the iterative style of an analyst's work when you want to spin data in different formats quickly. Moreover, this option is poorly suited for interactive reports with filters, etc.

3.2.2 Collaboration and co-operation

In most cases, we will want to not only look at notebooks but also run Jupyter notebooks, modify the code, and see how it works. Running the code requires computing resources. We have two options for this:

1. Install a Python environment on your computer and use the Jupyter/JupyterLab interface in a browser or Visual Studio Code.

2. Run in an online cloud environment using Binder, GitHub Codespaces, or other cloud environments.

In the first case, we have full control over the environment, files, and computing resources, but we will have to spend some time on installation. In the second case, we will be using someone else's computing resources, and most likely, the amount of free resources available to us will be limited. But we won't need to install any software, and you can get up and running in a matter of minutes.

3.3 Data Security

Jupyter provides a powerful and flexible environment for working with data. Still, it's also important to remember to protect your data, model, and environment from unauthorized access and tampering.

3.3.1 Protecting Sensitive Data

Before sharing your notebook, you need to make sure that all sensitive information is erased and replaced with anonymized data. Sensitive information may include any personal identifiers such as names, addresses, or social security numbers.

A VPN is also very suitable for data safety when exchanging confidential data. It can be used to encrypt your Internet connection, making it difficult for unauthorized persons to intercept or view your data.

Every employee should take a course on the importance of data security. But also, the person who sends the data must make sure that the recipient remembers all the rules about working with sensitive data

4 Methodology

4.1 Used data description

4.1.1 Data Sources

This study utilizes data from several internal Pipedrive sources, classified as private data, and made read-only to all accounts. The main data sources include the data shown in Table 1.

Table 1. S*chemas that are accessible to all accounts*

Schema naming	Schema description
dw_<business_area>	Cleansed and transformed data specific to a particular business area
dwa_<microservice_name>	Annotated data, primarily cleansed, with no custom transformations.
dwmodel_<business_area>	Analytical business models are designed to answer a group of questions specific to a particular business area.
dwaggr and report	Data aggregations and customizable reports.

Data is accessed through the Data Steward Schema Catalog interface. Data Steward is a simple web app for browsing, searching, and editing our Data Warehouse's metadata - the data about databases, schemas, tables, columns, and their comments.

The current version of Steward focuses on the documentation of the tables in the data warehouse and encourages analysts and data owners to keep it updated. For data queries, we use Jupyter Notebooks with Spark support or AWS Athena to run SQL queries. Sometimes, it's important to understand that Spark and Athena are not databases but query engines. They both use the same data stored in the cloud (AWS S3).

4.1.2 Data sampling

Data that meet the following criteria are selected for analysis:

- Dates and time periods identified for analysis (e.g., last quarter or year).
- Restrictions on specific business areas or microservices affecting the problem under investigation.
- Filtering of data by activity, such as excluding inactive records or records that do not meet certain criteria (e.g., HTTP response codes between 200 and 299).

In order to conduct a comprehensive analysis, it is important to select the data that will be included carefully. The analysis can be targeted and focused by considering various criteria, such as specific time periods or business areas. For example, trends and patterns within that specific period can be identified by narrowing down the time frame to the last quarter or year. [15]

Additionally, restrictions on specific business areas or microservices can provide further insight into the problem under investigation. By excluding unrelated or irrelevant data, the analysis can be streamlined, and the focus can be placed on the most important areas. [10]

Furthermore, filtering the data by activity can also be beneficial. The analysis can be more accurate and meaningful by excluding inactive records or records that do not meet certain criteria. For example, excluding data with HTTP response codes between 200 and 299 ensures that only successful responses are included, providing a more reliable dataset.

4.1.3 Data pre-processing

The data are subjected to the following pre-processing steps before analysis begins:

- Data cleaning: Removal or correcting incorrect, incomplete, or anomalous values.

- Missing value processing: Filling in missing values in the data or excluding them, depending on the context and relevance to the study.
- Data Transformation: Transforming data into a format suitable for analysis, including changing data types, normalizing, aggregating, or creating new variables from existing data.

This data sampling and pre-processing approach provide a sound basis for further data analysis and visualization within the study. [13] The main goal here is to prepare the data in such a way that it can be used effectively to answer the research questions posed in the study while ensuring that confidentiality and data security policies are respected.

4.2 Metrics Development

4.2.1 Data visualization

Visualisation plays a key role in interpreting the results of the analysis. The following visualization tools and techniques are used in this paper:

- Plotly: The main tool for creating interactive graphs and charts. Plotly allows the construction of dynamic visualizations that users can explore in real-time. [14]

In Jupyter notebooks using the PySpark kernel, various types of automatic visualizations for SQL queries are available, including tables, pie charts, line charts, area charts, and bar charts. This enables you to analyze data quickly and efficiently without diving into each visualization's complex programming details.

In addition, Jupyter Notebooks include magic commands such as %%sql for executing SQL queries and %%spark for executing code in Spark. These commands make it easy to switch between different code execution contexts and allow to leverage different analytical and visualization tools in the same workspace. In my work, I am gonna use both of these commands.

4.2.2 Identifying key metrics

The selection of key metrics related to the most important aspects of performance is one of the requirements for successful customer service performance and quality analysis in a firm. One such important metric is the response time for requests from customers with more than 15+ seaters, measured as the 99th percentile (p_99) of response time for requests.

P_99 response time is an observation that says all requests are handled within the stated time or less 99% of the time. It's a very important metric in customer service to understand the maximum latency experienced by most of the users.

4.2.3 Methods of calculating metrics

After defining the need to calculate the P_99 response time metrics based on log data, the actual process involves several steps implemented in Jupyter Notebooks using SQL queries and Python code for visualization. The pseudocode breakdown for each section in Figures 1-7 explains what happens and why each step is necessary:

```
%%sql
CREATE OR REPLACE TEMP VIEW barista_7d_logs AS
SELECT
  request_dt,
  key_company,
  key_baristaservice,
  key_httpmethod,
  uri,
  duration_total_ms
FROM dw.f_baristarequest
WHERE request_dt BETWEEN '2024-04-01' AND '2024-05-01'
  AND key_baristarequestauth = 'app'
  AND key_httpresponsecode BETWEEN 200 AND 299
  AND key_routingdirection IN ('--', 'none')
  AND key_company IN (
    SELECT key_company FROM dwmodel.d_company WHERE
total_seat_count >= 15)
```

Figure 1. *SQL of create a temporary view for raw log data (pseudocode)*

The initial query in Figure 1 generates a view underlaid dataset, barista_7d_logs, filtering logs within the set of dates and criteria. It goes on to select data only for requests that come out successful, within the range of HTTP status 200 to 299, and it omits the inter-region routing. It gives preference to important company interactions where an amount of seat count is reached to give room for activities related to bigger companies—15 or more.

```
%%sql
CREATE OR REPLACE TEMP VIEW barista_7d_aggr AS
SELECT
  DATE_FORMAT(request_dt, 'w') AS week,
  APPROX_PERCENTILE(t.duration_total_ms, 0.99, 300) AS
overall_p99
FROM barista_7d_logs AS t
GROUP BY 1
ORDER BY 1 ASC
```

Figure 2. *SQL of aggregate data for weekly performance (pseudocode)*

The query in Figure 2, executed after log view creation, performs data aggregation at the weekly level and calculates the 99th percentile of response times (overall_p99) for each week. In this step, we will examine the weekly performance to notice whether there are any specific weeks in the year when response times peak due to increased loads and other operational issues.

```
%%sql
CREATE OR REPLACE TEMP VIEW
barista_14d_slow_service_api_15plusseats AS
SELECT
  COUNT(*) AS total_req,
  t.key_baristaservice,
  t.uri,
  t.key_httpmethod,
  APPROX_PERCENTILE(t.duration_total_ms, 0.99, 300) AS
overall_p99
FROM barista_7d_logs AS t
GROUP BY 2, 3, 4
```

Figure 3. *SQL of create a temporary view to filter and calculate metrics (pseudocode)*

The next query in Figure 3 creates a temporary view, aggregating information of the log table: counts the total requests—total_req and looks for the 99th percentile of response time—overall_p99 for each service, URI, and HTTP method combination. This step is quite important, as it focuses on the optimization of the longest response times of endpoints.

```
%%sql
CACHE TABLE barista_14d_slow_service_api_15plusseats
```

Figure 4. *SQL of caching the temporary view (pseudocode)*

Caching the table that is shown in Figure 4 in memory helps improve performance for any other query that further accesses this data. This works great in a multi-query setting with gigantic datasets, preventing repetitive data fetching from the disk.

```
%%sql
CREATE OR REPLACE TEMP VIEW top_slow_apis_activities AS
SELECT *
FROM barista_14d_slow_service_api_15plusseats AS t
WHERE t.uri = '/api/v1/activities'
   OR t.uri = '/api/v1/activities/:id'
   OR t.uri = '/api/v1/activitytypes/'
ORDER BY total_req DESC
LIMIT 120
```

Figure 5. *SQL of create a temporary view for specific endpoints (pseudocode)*

This code in Figure 5 filters the previously collected data to represent only selected API endpoints that would have been related to "activities." The idea is to discover the important endpoints users interact with at this step and, from that data, extract the response times so this information can later be visualized.

```
%%sql -o df_activities
SELECT * FROM top_slow_apis_activities
```

Figure 6. *SQL of fetching and displaying the filtered data (pseudocode)*

This query in Figure 6 pulls from the view created in the previous step. In fact, the query itself is a dataset ready for use in an analysis with visuals, which indicates the specific endpoints that may be causing delays.

```
%%local
import plotly.express as px

# Assuming df_activities is the DataFrame containing your SQL
query results
fig = px.bar(df_activities, x='uri', y='overall_p99',
color='key_httpmethod', title='API Endpoint Latency (P99) for
Activities',
             labels={'overall_p99': 'Latency (ms)', 'uri':
'URI'}, barmode='group')
fig.update_layout(xaxis_title='API Endpoint',
                  yaxis_title='P99 Latency (ms)',
                  legend_title='HTTP Method')
fig.show()
```

Figure 7. *SQL of visualizing the data using plotly (pseudocode)*

This final step, which is shown in Figure 7, visualizes the data with a barchart implemented using Plotly, giving an interactive and clear representation of latency metrics for various API endpoints and methods. Visualization helps stakeholders quickly realize performance bottlenecks by method and endpoint, thus contributing to making more informed decisions regarding performance improvements.

4.3 Guide Developing

In order to produce a detailed guide on integrating metrics into Jupyter Notebooks for Pipedrive staff, there are several important stages to take into account. These stages are designed to guarantee that all employees can grasp the fundamental ideas and also gain the skills needed to incorporate and assess metrics independently, thus enhancing overall performance.

4.3.1 Instruction on Setting Up the Environment

Proper setup is the fundamental and essential preparatory work that creates the necessary foundation for all subsequent activities related to data analysis and metric development to take place and progress. It is of utmost importance to have an accurately arranged and organized environment in order to ensure the smooth execution and success of various subsequent processes such as data fetching, processing, and visualization. Without a meticulously configured environment, the effectiveness of these subsequent steps can be greatly hindered and compromised. Therefore, it is crucial to prioritize and invest ample time and effort in the initial setup phase to lay a solid groundwork for seamless and fruitful data analysis endeavors. [15]

4.3.2 Data Access and Preparation

This section is of utmost importance as it provides users with the necessary knowledge and skills to establish connections between Jupyter Notebooks and an extensive range of data sources, including SQL databases, cloud storage systems, and APIs. Gaining a profound understanding of the procedures involved in accessing data marks the initial and pivotal phase in achieving the capability to conduct comprehensive and significant analyses efficiently and effectively. By acquiring proficiency in data access techniques, users will be equipped with the essential tools to extract valuable insights. The ability to connect to diverse data sources empowers analysts to access a vast array of information, ensuring a wide-ranging and comprehensive analysis. With this comprehensive understanding, users will be poised to uncover valuable insights and accelerate decision-making, paving the way for informed data-driven strategies. [16]

4.3.3 Visualization Techniques

Provide an extensive overview and comprehensive analysis of popular Python libraries like Matplotlib and Plotly, which are widely recognized and highly utilized for creating visually appealing and informative visualizations in Jupyter Notebooks. The aim is to empower users with an in-depth understanding of each library's strengths, capabilities, and unique features, enabling them to make informed decisions when selecting the most appropriate tool that aligns perfectly with their specific visualization requirements. By delving into the intricacies of these libraries, users will gain invaluable insights into the various use cases and scenarios in which each library excels, thus ensuring each visualization endeavor achieves remarkable success.

4.3.4 Best Practices and Troubleshooting

Best practices play a pivotal role in guiding users on how to make the most of Jupyter Notebooks, ensuring optimal utilization of its capabilities. These practices enable users to execute commands efficiently and acquaint them with methodologies that can significantly enhance the efficiency and reliability of their work. By adhering to these best practices, teams can achieve standardization in their processes, fostering consistency in the quality of work across the entire organization. [15]

4.3.5 Additional Resources

Including a comprehensive section on additional resources is crucial for fostering an environment of continuous learning and development within the data science and technology fields. These fields evolve rapidly, and it is imperative for professionals to keep up with the latest tools, techniques, and best practices in order to maintain a competitive edge in the industry. By providing a diverse range of resources, this section not only encourages employees to expand their knowledge beyond the basics covered in the guide but also empowers them to enhance their skills constantly. [17]

5 Results

5.1 P_99 metric

The graph that was made shows the API latency in the 99th percentile (P99) for different HTTP methods and specific endpoints. This graph helps analyze the API response time, which is important for optimizing performance and improving user experience. By examining the latency data for each request, we can gather valuable insights about the efficiency of the API and identify areas that need improvement. [10]

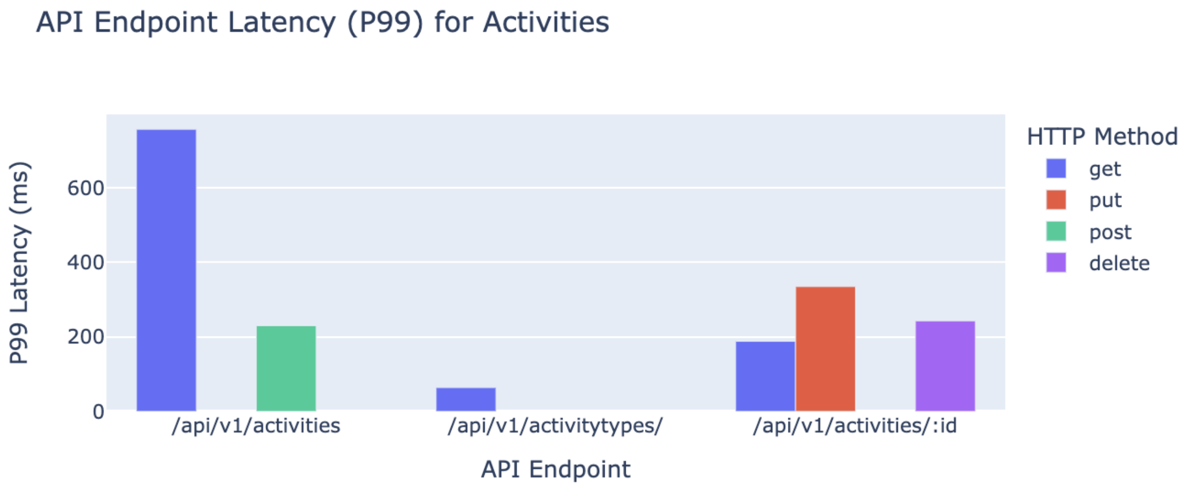

Figure 8. *API latency for activities endpoint*

This chart in Figure 8 can reveal the following insights and provide information on:

1. /api/v1/activities:
    a. GET: The latency is 757 ms, which is the highest latency among all the requests shown. This indicates that GET requests on this endpoint may require optimization. Further investigation is needed to understand why these particular requests are taking longer than expected. It may be necessary to optimize the database or cache frequent requests.
    b. POST: On the other hand, the latency of 229 ms indicates that the response time for POST requests is much faster compared to GET requests. This

positive finding shows that the API performs well in handling POST requests on this endpoint.

2. /api/v1/activitytypes/:
   a. GET: The latency of 64 ms is the lowest in the graph, which showcases the high efficiency of processing requests to this endpoint. The API is able to handle GET requests on this endpoint swiftly, resulting in a superior user experience. On the other hand, it can also indicate that users are not performing actions that require using this specific endpoint.
3. /api/v1/activities/:id:
   a. GET: Latency recorded at 189 ms, indicating reasonable performance for retrieving specific activity details.
   b. PUT: We notice a delay of 335 ms for PUT requests on this endpoint, indicating the need to check the data processing when updating existing records. Additional measures must be taken to ensure the data update process is efficient and the latency is reduced.
   c. DELETE: Records a latency of 236 ms, pointing to a relatively efficient deletion process, but still with room for improvement, especially when compared to the POST requests at the main activities endpoint.

Based on this data, the following steps can be considered:

1. Optimise GET requests for /api/v1/activities: Given the high latency observed in these GET requests, it is worth analyzing why these specific requests are taking longer than desired. It may be necessary to optimize the database or cache frequent requests to improve the overall performance.
2. Monitoring and analyzing PUT requests: As PUT requests have a significant latency, ensuring that the data update process is efficient is crucial. Regular monitoring and analyzing the latency for these requests will help identify areas that can be improved to enhance overall performance.
3. Maintaining performance for /api/v1/activitytypes/: While the low latency observed for this endpoint is a positive indicator, it is important to continue monitoring to sustain this level of performance.

5.2 Knowledge-Sharing

The knowledge-sharing document developed for this thesis is an elaborative manual customed for the Pipedrive team in detail regarding the use of Jupyter Notebooks in every aspect. It started by introducing the user to Jupyter Notebooks and its relevance to handling complicated data in a presentable, clear, and right manner. Afterward, I went through installation and basic/advanced operation features in Jupyter to ensure that the user was well-versed in the system.

It covers practical applications, with step-by-step tutorials on how data is fetched, processed, and analyzed using SQL and Python in the Jupyter environment. In the practical parts, I have prepared sections about how to write effective SQL queries to get data from our internal systems, and later, the following parts are about using Python for wrangling and on-the-fly visualization with Plotly. As an example, I used a simple query to find the top 100 companies with the most updated activities between '2024-04-01' and '2024-05-01'. In the end, this is an example of a metric that can be used to identify the number of updated activities. This is a tangible outcome that users can achieve when they follow the instructions in the document.

In this final presentation of the metric in Figure 9, it is to be noted that specific identifiers, such as company_id, have been carefully omitted due to reasons of privacy standards.

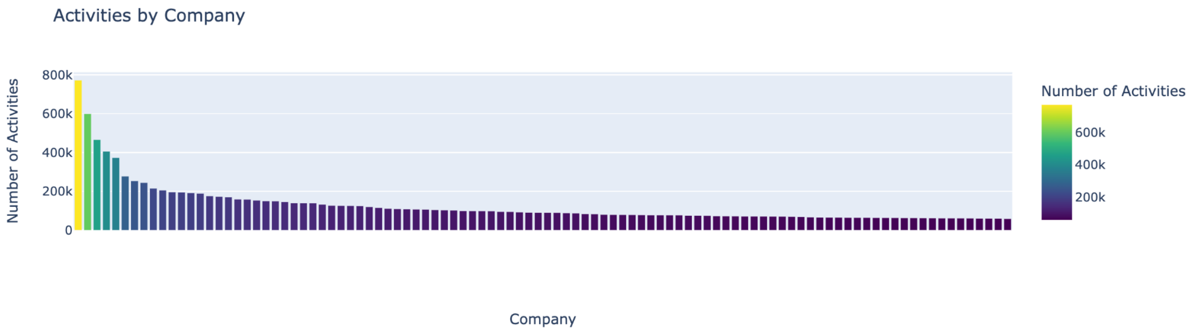

Figure 9. *Top 100 companies with the most updated activities*

Besides, the document gives performance-optimizing tips and tips on how to solve some problems that may be faced in the course of using Jupyter Notebooks, hence being a guide for users at both beginner and expert levels. More resources and additional reading materials linked in the guide shall further enhance the team's skills and competencies in the field of data science and analytics.

6 Summary

This thesis on the use of Jupyter Notebooks for visualizing product and engineering metrics at Pipedrive really helps bring to the fore just how one can increase their data analysis and visualization capability manyfold in a business setting. It outlined a thorough approach of including and using Jupyter Notebooks from the beginning, directly from the environment setup through in-depth metric analysis, such as making one critical P_99 latency metric.

The research results gained great acceptance within Pipedrive, and the built metrics answered how fast the API key response must be to maximize system performance. With these metrics visualized using Plotly and other similar tools, the team was readily able to identify and tackle potential performance bottlenecks. Furthermore, the comments of the team not only validated the importance of the work but also brought out ideas for future enhancements. Other needed enhancements were a timeline graph that monitored the changes in the last six months, as there was intense enthusiasm for building the analytical skills that had been inducted.

The manual developed to support the thesis gave a structured walkthrough for Pipedrive employees on how to approach using Jupyter Notebooks in their data analysis process. It included data access and preparation practices and advanced visualization best practices, which equipped employees to leverage working with Jupyter Notebooks.

Evidently, if the metrics were successfully implemented and positive feedback was received in relation to the pilot test phase, Jupyter Notebooks can be regarded as a tool that may revolutionize practice in the management of data at Pipedrive. In general, this research has enriched the scholarly world with the implementation of Jupyter Notebooks in business analytics and the uplift of my Pipedrives team by enabling better product and engineering efforts. The outlook for this project in the future involves the expansion of the metric system to cover much more comprehensive time-series analyses, as well as the incorporation of other predictive analytic features that would further boost decision processes at Pipedrive.

Appendix 1 – Non-exclusive licence for reproduction and publication of a graduation thesis[1]

I Maria Kapitonova

1. Grant Tallinn University of Technology free licence (non-exclusive licence) for my thesis ”Visualizing product and engineering metrics in Jupyter notebooks at Pipedrive”, supervised by Inna Švartsman
    1.1. to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright;
    1.2. to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.
2. I am aware that the author also retains the rights specified in clause 1 of the non-exclusive licence.
3. I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

18.05.2024

[1] The non-exclusive licence is not valid during the validity of access restriction indicated in the student's application for restriction on access to the graduation thesis that has been signed by the school's dean, except in case of the university's right to reproduce the thesis for preservation purposes only. If a graduation thesis is based on the joint creative activity of two or more persons and the co-author(s) has/have not granted, by the set deadline, the student defending his/her graduation thesis consent to reproduce and publish the graduation thesis in compliance with clauses 1.1 and 1.2 of the non-exclusive licence, the non-exclusive license shall not be valid for the period.

Bibliography

[1] L. Murta, "A Large-scale Study about Quality and Reproducibility of Jupyter Notebooks." [Online]. Available: https://leomurta.github.io/papers/pimentel2019a.pdf (visited on 20.04.2024).

[2] "The Challenge of Autograding in Jupyter Notebooks," Fabian Monrose's Website. [Online]. Available: https://fabianmonrose.github.io/papers/autograding23.pdf (visited on 10.05.2024).

[3] X. Huseynov, "Introduction to Data Science with Apache Zeppelin," ICT Conference on Big Data. [Online]. Available: https://ict.az/uploads/konfrans/biq_data/1-5_Xalid_Huseynov_Introduction_to_Data_Science_with_Apache_Zeppelin.pdf (visited on 20.04.2024).

[4] "Apache Zeppelin Documentation v0.6.0," Apache Zeppelin. [Online]. Available: https://zeppelin.apache.org/docs/0.6.0/ (visited on 20.04.2024).

[5] "The Most Important Apache Zeppelin Use Cases," Dremio Wiki. [Online]. Available: https://www.dremio.com/wiki/apache-zeppelin/#:~:text=The%20Most%20Important%20Apache%20Zeppelin%20Use%20Cases&text=It%20supports%20various%20libraries%20and,analysis%2C%20and%20sensor%20data%20analysis. (visited on 20.04.2024).

[6] "What is a Notebook?" Jupyter Documentation. [Online]. Available: https://docs.jupyter.org/en/latest/#what-is-a-notebook (visited on 02.05.2024).

[7] "What is the Jupyter Notebook?" Jupyter Notebook Documentation. [Online]. Available: https://jupyter-notebook.readthedocs.io/en/latest/examples/Notebook/What%20is%20the%20Jupyter%20Notebook.html (visited on 02.05.2024).

[8] "Jupyter Notebook Documentation," Jupyter Notebook. [Online]. Available: https://jupyter-notebook.readthedocs.io/en/latest/notebook.html (visited on 02.05.2024).

[9] "Advances in Database Technology," ScienceDirect. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167739X21003976 (visited on 10.05.2024).

[10] "Exploring Time Series Databases," UPC Commons. [Online]. Available: https://upcommons.upc.edu/bitstream/handle/2117/341629/A?sequence=3 (visited on 13.05.2024).

[11] "An Introduction to Data Visualization Tools and Techniques in Various Domains," ResearchGate. [Online]. Available: https://www.researchgate.net/publication/370593444_An_Introduction_to_Data_Visualization_Tools_and_Techniques_in_Various_Domains (visited on 20.04.2024).

[12] "About Pipedrive," Pipedrive. [Online]. Available: https://www.pipedrive.com/en/about (visited on 20.04.2024).

[13] "Current Trends in Knowledge Engineering," ScienceDirect. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0165993620302740 (visited on 12.05.2024).

[14] "Getting Started with Plotly," Plotly. [Online]. Available: https://plotly.com/python/getting-started/ (visited on 12.05.2024).

[15] "How Data Sampling Impacts Business Decisions," Hoa Sen University Library. [Online]. Available: https://thuvienso.hoasen.edu.vn/bitstream/handle/123456789/7391/Contents.pdf?sequence=4 (visited on 13.05.2024).

[16] L. Anthonysamy, "Self-regulated Learning Strategies in Higher Education: Fostering Digital Literacy for Sustainable Lifelong Learning," ResearchGate. [Online]. Available: https://www.researchgate.net/profile/Lilian-Anthonysamy/publication/341343124_Self-regulated_learning_strategies_in_higher_education_Fostering_digital_literacy_for_sustainable_lifelong_learning/links/5ebb8a99a6fdcc90d672411a/Self-regulated-learning-strategies-in-higher-education-Fostering-digital-literacy-for-sustainable-lifelong-learning.pdf (visited on 13.05.2024).

[17] "How Jupyter Notebooks Streamline Data Analysis," Google Books. [Online]. Available: https://books.google.ee/books?hl=en&lr=&id=tqTsDwAAQBAJ&oi=fnd&pg=PP1&dq=Restrictions+on+specific+business+areas+can+streamline+analysis+in+Jupyter+Notebooks.&ots=3yBetx5L-K&sig=GYVpNtzcWXM7NwsbyIVPsTFNi5w&redir_esc=y#v=onepage&q&f=false (visited on 13.05.2024).