Executive Summary: Technical Design Choices

1. Introduction

The goal of this project is to develop a reliable pipeline that automatically generates high-quality reports in markdown format. These reports provide insightful analysis of BMW sales data spanning 2020 to 2024, leveraging the power of Large Language Models (LLMs) for natural language generation and data interpretation.

---

2. Approach and Design Choices

2.1. LLM Selection and Autonomy Level

For natural language generation, I chose **Google Gemini 2.5 Flash**. It strikes a good balance between model capabilities and accessibility, with a free usage quota that fits this project's scope nicely.

One of the key design decisions was determining how much autonomy the LLM should have within the pipeline. I considered **three levels**:

1. **Full autonomy:** The LLM would generate Python code for data processing, plotting, and report compilation entirely on its own.
2. **Partial autonomy with tools:** The LLM could call predefined tools or functions to assist report creation.
3. **Limited autonomy:** The LLM focuses solely on generating textual reports, while data processing and plot generation remain under explicit developer control.

I settled on the **third approach** to ensure that the report outputs remain consistent, accurate, and of high quality. By explicitly managing data processing and visualization, I can guarantee that the charts are meaningful and that the LLM's analysis is grounded in solid, reliable inputs.

2.2. Modular Multi-Step LLM Invocation

Rather than handing the entire report generation task over to the LLM in one go, I broke the interaction into **five focused stages**. This modular approach helps improve the clarity and quality of the final report by providing the LLM with targeted context at each step:

1. **Sales Trend Analysis:** Preprocess sales data to extract overall sales trends over time and regional sales patterns, then feed this to the LLM for commentary.
2. **Model Performance Over Time:** Analyze model-level sales data trends by year, enabling the LLM to provide targeted insights.
3. **Regional Model Performance:** Further refine analysis by combining regional and model data for deeper insights.

4. **Key Sales Drivers:** Conduct a **Pearson correlation analysis** to identify factors most strongly related to sales volume, which the LLM then interprets.
5. **Report Generation:** The LLM consolidates all prior analyses into a cohesive, business-friendly markdown report.

This staged methodology allows for richer, more accurate interpretations while maintaining control over each analysis phase.

2.3. Visualization Design

Since much of the data is time-series based, I focused on line graphs to clearly depict trends and changes over time. The report itself is structured to flow naturally, from broad overviews like total sales trends, moving toward detailed breakdowns such as regional sales and model-specific performance, and concluding with an analysis of key sales drivers. This progression helps readers follow the story logically and intuitively.

2.4. Modular Code Structure

The pipeline is designed with modularity in mind, supporting maintainability and future extension. The directory layout is shown in the Readme file.

---

3. Evaluation Framework

Quality assurance is an important part of the project. I implemented **unit tests**, located in the **tests/** folder, that cover the critical components:

- **Data loader functionality:** Ensure the BMW sales dataset is loaded and preprocessed correctly.
- **Plotting functions:** Verify that plotting functions generate the expected visualizations without errors.

This automated testing provides confidence that the data pipeline and visualization modules behave as intended, which supports the generation of accurate, trustworthy reports.

---

4. Conclusion

This approach balances the strengths of modern LLMs with developer oversight to produce accurate, insightful, and polished reports. The multi-step LLM invocation strategy encourages detailed and context-aware analysis, while the modular codebase and testing framework promote maintainability and robustness. Altogether, this pipeline reliably delivers meaningful markdown reports that can effectively support business decision-making.