**Assignment 2**

**Read the article**[**Software Engineering for Machine Learning: Characterizing and Detecting Mismatch in Machine-Learning Systems**](https://insights.sei.cmu.edu/blog/software-engineering-for-machine-learning-characterizing-and-detecting-mismatch-in-machine-learning-systems/)**by Lewis and Ozkaya (2021). Give a small summary of this article in which you describe the different types of mismatches the authors identify. Try to come up with concrete examples of these mismatches from your own projects and/or experience.**

The article is discussed about the importance of mismatch detection in machine learning. The different types of mismatches are in the following:

1. Computing-resource mismatch: This refers to poor system performance when the required computing resources for executing the ML model are not available in the production environment. For example, if a model requires significant computational power but the production environment lacks sufficient resources, it may lead to degraded performance or even system failures.
2. Data-distribution mismatch: ML models heavily depend on training data, and if the training data differs significantly from the production data, the model's accuracy can be compromised. For instance, if an ML model is trained on data collected from a specific region or time period but is deployed in a different region or time period, its performance may suffer.
3. API mismatch: ML components often interact with other system components through APIs. If the ML component expects different inputs or outputs than what is provided by the integrated system, it can lead to the need for generating excessive glue code or integration difficulties.
4. Test-data mismatch: Software engineers may face challenges in properly testing ML components due to limited access to test data or a lack of understanding of the component itself. This can impede the verification and validation process and potentially introduce errors or bugs.
5. Monitoring mismatch: ML-enabled systems require runtime monitoring of the ML components and relevant metrics such as model accuracy. However, the monitoring tools in the production environment may not collect the necessary ML-specific metrics, hindering the ability to assess the performance and health of the ML models.

The examples:

Imagin in a project where an ML model was developed to predict customer churn in a telecommunications company, and a **data-distribution mismatch** occurred. The reason is that, the model was trained on historical customer data collected from a specific region, but during deployment, it was used to make predictions for customers from different regions. Due to variations in customer behavior and preferences across regions, the model's accuracy decreased significantly, resulting in ineffective churn prediction and customer retention efforts.

Another example relates to API mismatch. In a project involving the integration of an ML-based recommendation system into an e-commerce platform, the ML component expected a specific data format and structure as input, while the existing system provided different data representations. This discrepancy required the development of custom data transformation code to bridge the gap between the ML component's requirements and the system's data format.

These examples illustrate the importance of addressing mismatches in ML-enabled systems to ensure optimal performance and accurate results. Bottom of Form

**Read the article**[**Tackling Collaboration Challenges in the Development of ML-Enabled Systems**](https://insights.sei.cmu.edu/blog/tackling-collaboration-challenges-in-the-development-of-ml-enabled-systems/)**by Lewis (2023) and give a small summary of it. How do you envision your cooperation as (future) data scientist with software engineers? In what way will you make your code scalable, readable and maintainable? If you have ever put code in production, please describe your experiences in doing so.**

‘*What are the collaboration points and corresponding challenges between data scientists and engineers?*’ This is the main challenge of this article.

During research the interviews conducted with numerous individuals engaged in the development of ML-enabled systems, the researchers identified three key collaboration points between data scientists and engineers: requirements and planning, training data, and product-model integration.

As a data scientist, collaboration with software engineers is crucial to ensure the successful implementation of data-driven solutions. Both roles bring complementary skills to the table. Data scientists focus on analyzing and interpreting data, building models, and extracting insights, while software engineers specialize in designing, developing, and maintaining robust and scalable software systems.

To ensure effective cooperation, data scientists and software engineers need to communicate and understand each other's requirements and constraints. Data scientists should provide clear documentation and specifications for their models, algorithms, and data processing steps. They should also work closely with software engineers to understand the engineering constraints and considerations, such as performance, scalability, and security.

I, as a data scientist, should try to make the code scalable, readable, and maintainable. Some aspects to do this are the following:

* Breaking down code into smaller functions or modules, because this method promotes reusability and makes it easier to understand, test, and maintain.
* Providing clear documentation for the code, including function descriptions, and input/output specifications, helps other team members understand and use the code effectively.
* Following established coding standards
* Optimizing code for efficiency and speed is essential